令和元年度　修士論文

CNN による最適画面内予測モード推定を利用した動画像符号化

早稲田大学 基幹理工学研究科 情報理工・情報通信専攻

5118F111-8

横山　怜汰

指導　甲藤二郎 教授
研究指導名　画像情報研究

2020 年 1 月 29 日

指導教授印	受付印

目次

第 1 章 序論

1.1 はじめに

近年、スマートフォンやタブレットといった携帯端末の普及により、多くの動画が生成および視聴されるようになってきている。なかでも YouTube や Netflix といった動画配信サービスの拡大により、多くの動画がインターネットからストリーミングやダウンロードによって視聴されている[1]。そして、図 1.1 に示すように動画のトラフィック量は今後ますます増加することが予測されている[2]。一方で情報量の大きい 4K や 8K といった高精細な動画コンテンツも拡がりを見せている。このような背景で多くの動画の蓄積や伝送を行うために、動画の高効率な圧縮が求められる。

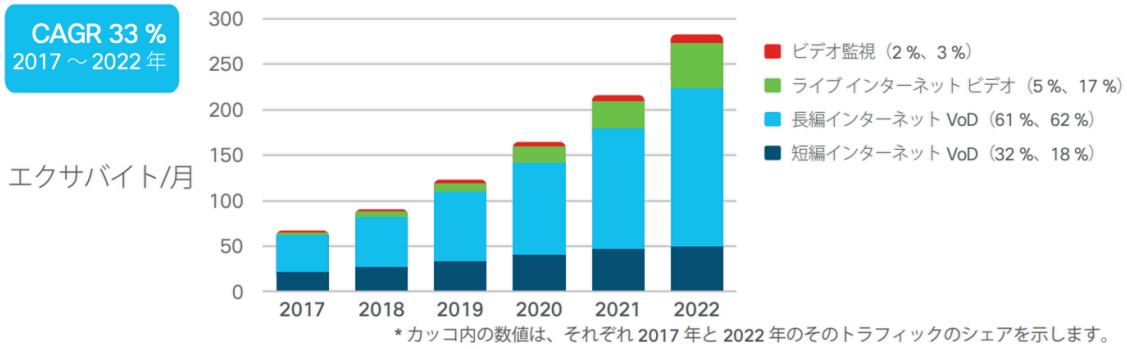

図 1.1　動画のトラフィック量の推移とその予測[2]

H.265/HEVC（High Efficiency Video Coding）は、JCT-VC（Joint Collaborative Team on Video Coding）により国際標準化された最新の動画像符号化規格である。HEVC では様々な要素技術を用いて高効率な動画像符号化を実現しており、画面内予測はその中で用いられている重要な要素技術の 1 つである。

1.2 研究目的

HEVC において、ピクチャはいくつかの Prediction Unit（PU）に分割される。そして、各 PU において符号量（bitRate）が少なくかつ符号化前の画像との歪み（Distortion）が小さくなる、すなわち Rate-Distortion（RD）最適となるような画面内予測モードが決定される。HEVC の参照ソフトウェアである HEVC Test Model（HM）

では、各モードの RD コストを計算することで、ほぼ RD 最適な画面内予測モードが選択できるが、計算量が大きい。そのため、より計算量の小さい Sum of Absolute Transform Difference（SATD）コストや Most Probable Mode（MPM）により RD コストの小さいモードを推定し、RD コスト計算対象のモード（RD モード）の数を絞り込む。本研究では、この RD モードの絞り込みの部分について、Convolutional Neural Network（CNN）を用いて改善し、より RD 最適な画面内予測モードを選択して符号化を行うことを目的とする。

1.3 本論文の構成

第 1 章では、本研究の背景や目的について述べてきた。第 2 章では、本研究に関連する基本的な技術について述べる。第 3 章では、本研究と密接に関わる HEVC における画面内予測について述べる。第 4 章では、CNN を用いて最適な画面内予測モード推定を行う手法について述べる。第 5 章では、提案手法について述べる。第 6 章では、提案手法の評価実験について述べる。第 7 章では、本論文の総括を述べる。

第 2 章 関連技術

本章では、本研究に関連する基本的な技術として、表色系、動画像符号化に関する基礎技術、動画像符号化標準の歴史、画質評価、Neural Network について述べる。

2.1 表色系

画像などにおいて色を表す体系のことを表色系という。一般にすべての色は互いに独立な 3 つの変数を用いて表せるとされていて、いくつかの表し方、すなわち表色系が提案されている。本節では、表色系として RGB と YCbCr について述べる。

2.1.1 RGB 表色系

RGB 表色系は、赤（R：Red）、緑（G：Green）、青（B：Blue）の 3 つの基本色を混ぜ合わせてさまざまな色を表現する。これは、色を感じ取る人間の目の細胞である錐体が、3 つの基本色に特に強く反応することに基づいている。RGB の 3 色は光の 3 原色と呼ばれていて、これらの色の光を重ねるほど明るい色が得られる。このように色を作り出す方式は加法混色と呼ばれ、ディスプレイなどで用いられている。[3]

2.1.2 YCbCr 表色系

YCbCr 表色系は、テレビ方式に関する標準の ITU-R（International Telecommunication Union - Radiocommunication Sector） BT.601 や BT.709 で定められた、輝度信号と 2 つの色差信号を用いて色を表現する表色系である。輝度信号（Y）と色差信号（Cb、Cr）は、RGB から変換式を用いて求められる。式(2.1)に BT.601 で定められた SDTV（Standard Definition TeleVision）での変換式を、式(2.2)に BT.709 で定められた HDTV（High Definition TeleVision）での変換式を示す。また、図 2.1 にデジタル画像を RGB と YCbCr の各成分に分解して可視化した図を示す。

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.1}$$

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.213 & 0.715 & 0.072 \\ -0.115 & -0.385 & 0.500 \\ 0.5000 & -0.454 & -0.046 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.2}$$

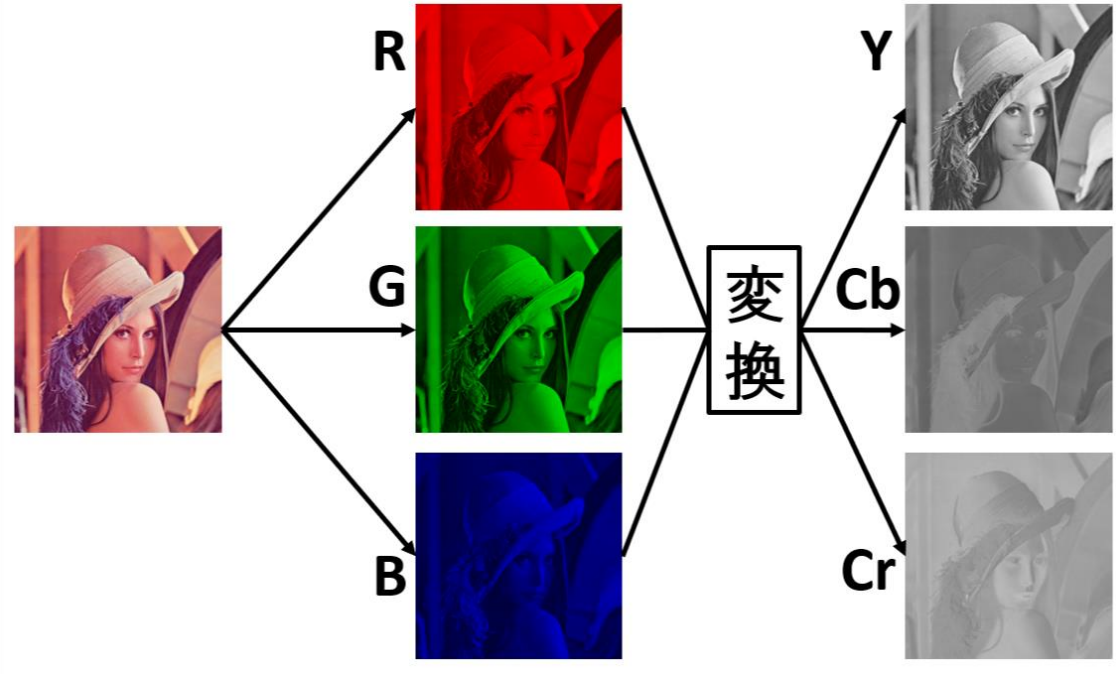

図 2.1　デジタル画像（Lena[4]）を RGB と YCbCr の各成分に分解して可視化した図

YCbCr 表色系と似た表色系として YUV や YIQ があるが、Y 成分の変換式は式(2.1)と同じで色差成分の変換式がそれぞれ異なる式になっている。[5] このように輝度信号と色差信号を用いる表色系はテレビ放送などの動画像標準で採用されている。その理由としては、輝度信号が色差信号と分離していることで次のようなメリットがあるためである。1 つ目は、輝度信号のみ再生して白黒画像を表示することができ、カラー放送と白黒放送の両方に対応できることである。2 つ目は、人間の目が輝度より色差の変化に気づきにくい性質を利用して色差信号を間引くことにより高効率な圧縮が可能になることである。色差信号について水平方向を半分に間引いたものを 4:2:2 フォーマット、水平垂直両方向を半分に間引いたものを 4:2:0 フォーマット、間引かないものを 4:4:4 フォーマットという。4:4:4 や 4:2:2 は主にコンテンツ制作や放送用の素材に使われ、4:2:0 は広く消費者向けのアプリケーションに使われている。図 2.2 に 4:4:4、4:2:2、4:2:0 フォーマットの違いを示す。[6]

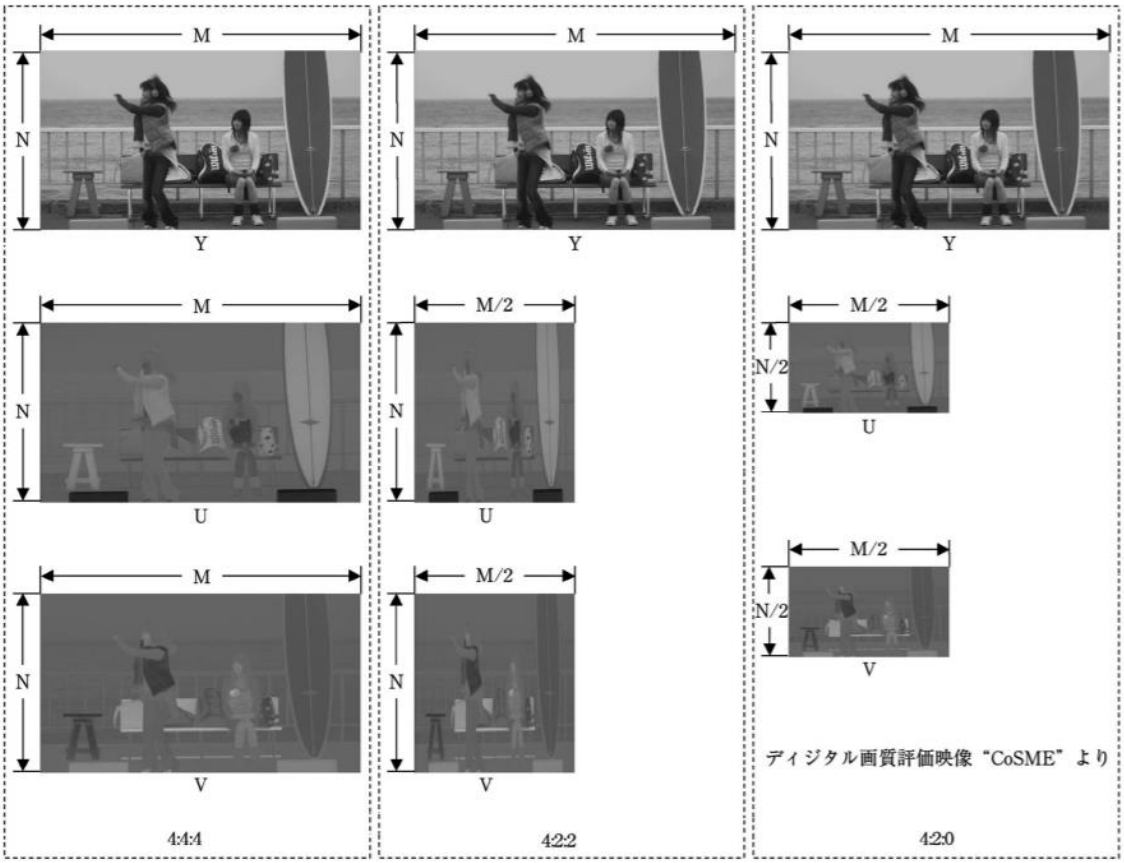

図 2.2　4:4:4、4:2:2、4:2:0 フォーマットの違い[6]

2.2 動画像符号化に関する基礎技術

本節では、動画像の符号化に関する基本的な技術について説明する。はじめに、アナログの画像情報をデジタルに変換する A/D 変換について説明する。そして、HEVC を含めた近年の動画像符号化で用いられている、画面内予測、画面間予測、直交変換、エントロピー符号化について説明する。

2.2.1 A/D 変換

カメラから得られる画像情報は、もともと情景を電気信号に変換したアナログ信号である。しかし、圧縮符号化はデジタル信号処理で行われるため、Analog to Digital（A/D）変換が行われる。A/D 変換には(1)標本化、(2)量子化、(3)符号化のステップがある。

(1)標本化では、アナログ信号から一定の時間間隔で標本を取り出す。このとき、入力のアナログ信号の最大周波数 2 倍以上の周波数で標本を取り出すようにする。これは、シャノン・染谷の標本化定理に基づいている。

(2)量子化では、取り出された標本の値を連続的な値から離散的な値にする。量子化のレベルをどれほどの細かさで刻むかによって信号の滑らかさが決まる。

(3)符号化では、量子化された数値を 0 と 1 の符号を用いたデジタル値にする。

図 2.3 に、A/D 変換の仕組みを示す。

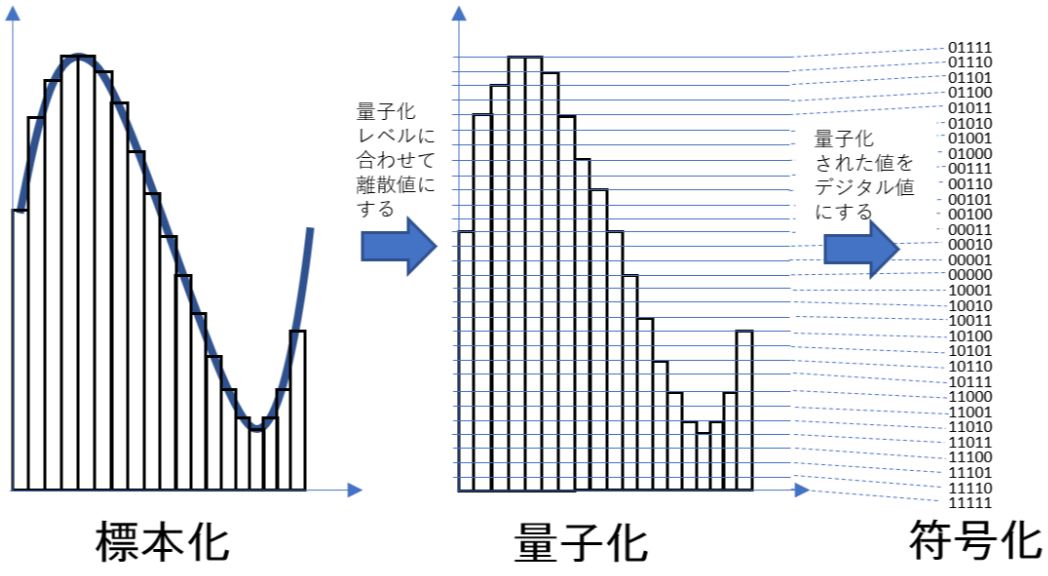

図 2.3　A/D 変換

2.2.2 画面内予測

画面内予測（Intra Prediction）とは、画素値が空間的に高い相関をもつことから周囲の画素を利用して行う予測のことである。例えば、横縞の柄の画像であれば左隣の画素値から予測を行えるため、その分情報量を削減できる。H.264/AVC（Advanced Video Coding）や HEVC といった近年の動画像符号化規格では、複数の画素からなるブロック単位で、いくつか用意された画面内予測モードから適したものを選択して画面内予測が行われる。図 2.4 に AVC における画面内予測モードを示す。[7] HEVC における画面内予測については 3 章で説明する。

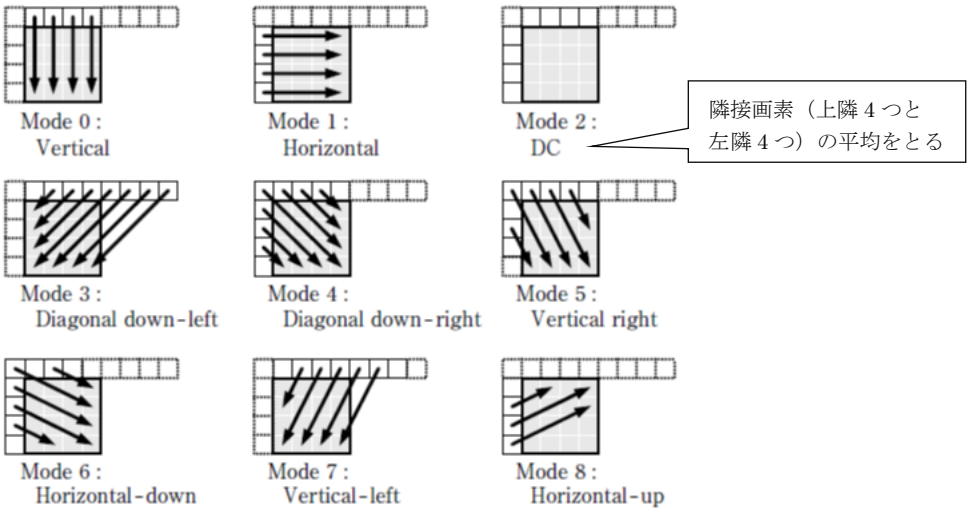

図 2.4　AVC における画面内予測モード[7]

2.2.3 画面間予測

画面間予測（Inter Prediction）とは、画素値が時間的に高い相関をもつことから前後付近のフレームの画素を利用して行う予測のことである。例えば、動きのないシーンの動画像であれば直前フレームの同位置の画素値から予測を行えるため、その分情報量を削減できる。一方、動きの大きなシーンでは直前フレームの同位置の画素値から単純に予測しようとしても当たらなくなる。しかし形状の変化しない物体が動くようなシーンの場合、物体の動きベクトルが分かれば直前フレームで物体があった位置の画素値から予測が可能となる。このような仕組みで行う予測を動き補償画面間予測という。図 2.5 に、単純な画面間予測と動き補償画面間予測の例を示す。同図を見ると、単純な画面間予測よりも動き補償画面間予測を用いたほうが、予測誤差が小さくなっていることが確認できる。さらに、

動き補償画面間予測を過去と未来の双方のフレームから行うものを双方向動き補償画面間予測という。双方向予測は、過去のフレームのみ用いる順方向予測と比べてより正確な予測が期待できるが、未来のフレームを用いるために本来のフレームの順序を入れ替える必要があり遅延が生じるという側面もある。

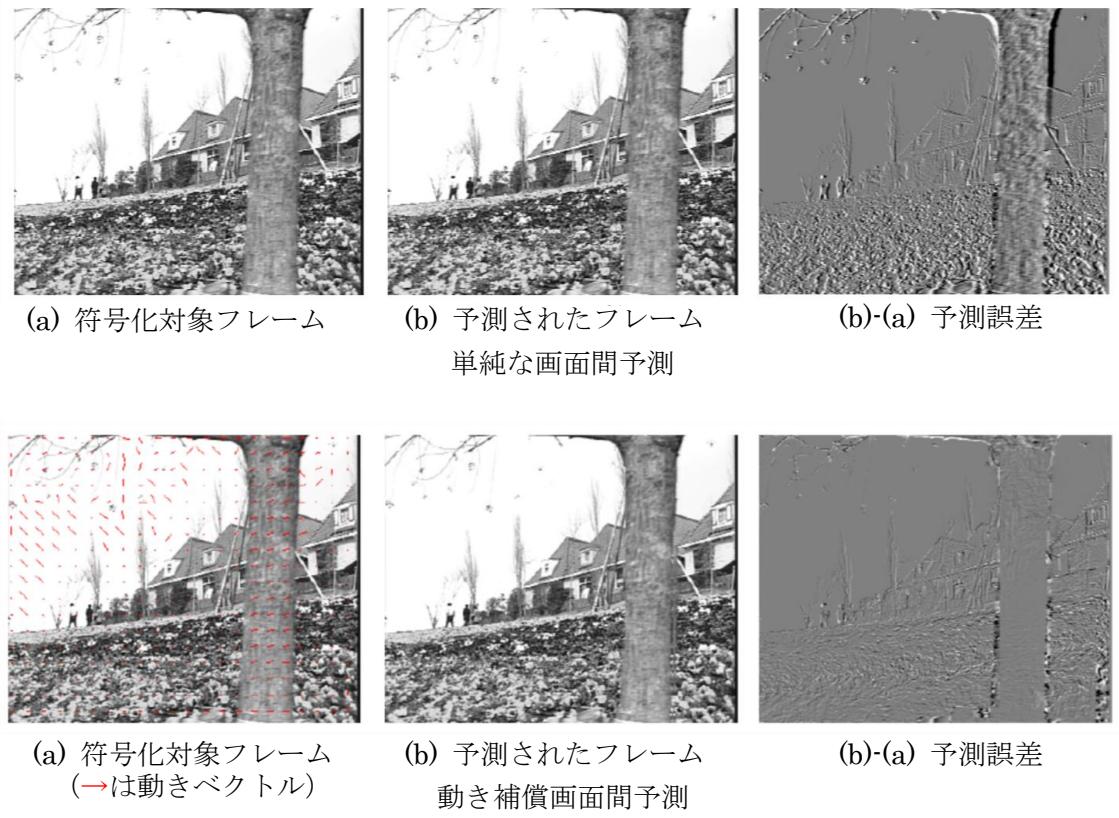

図 2.5　単純な画面間予測と動き補償画面間予測の例[8]

2.2.4 直交変換

画像信号の相関性を利用して直交変換を行い、特定の成分にパワーを集中させることで情報量を削減できる。画像信号は低周波成分にパワーが集中する傾向があることが知られているため、低周波成分へのパワーを集中させるのに優れた直交変換を用いることになる。このような低周波成分にパワーを集中させる変換は、人間の目が画像の高周波成分には鈍感であるといった特性があることからもよい変換方法といえる。そして、単に変換させるだけでなく、高周波成分は粗く低周波成分は細かく量子化することによって情報量を削減でき効率の良い圧縮が可能となる。

代表的な直交変換の方法としてはフーリエ変換があげられるが、複素数演算による計算コストが高く他の変換方法が求められる。

Hadamard（アダマール）変換は、変換行列の要素が+1 と-1 のみで構成されており、加減算のみで乗算の必要なく変換が行えるという簡単な変換手法である。演算が簡単である反面、低周波成分へのパワー集中度では他の変換方法に劣る。アダマール変換の基本変換行列を式(2.3)に、$2n$次の変換行列を式(2.4)に示す。式(2.3)の基本行列をもとに式(2.4)を再帰的に用いることで高次の変換行列が得られる。

$$H_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.3}$$

$$H_{2n} = \frac{1}{\sqrt{2}}\begin{bmatrix} H_n & H_n \\ H_n & -H_n \end{bmatrix} \tag{2.4}$$

なお、アダマール変換は逆変換も同じ変換行列で表される。[9]

離散コサイン変換（DCT：Discrete Cosine Transform）は、アダマール変換よりも低周波成分へのパワー集中度が高い変換である。$N \times N$の画像に対する DCT は式(2.5)、逆変換は式(2.6)で表される。

$$F(u,v) = \frac{2}{N}C(u)C(v)\sum_{j=0}^{N-1}\sum_{k=0}^{N-1} f(j,k)\cos\left[\frac{(2j+1)u\pi}{2N}\right]\cos\left[\frac{(2k+1)v\pi}{2N}\right] \tag{2.5}$$

$$f(j,k) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} C(u)C(v)F(u,v)\cos\left[\frac{(2j+1)u\pi}{2N}\right]\cos\left[\frac{(2k+1)v\pi}{2N}\right] \tag{2.6}$$

ただし、$f(j,k)$は画像信号、$F(u,v)$は DCT の係数を表し、

$$C(u), C(v) = \begin{cases} \dfrac{1}{\sqrt{2}}\ (u,v=0) \\ 1\ (u,v=1,2,\dots,N-1) \end{cases} \tag{2.7}$$

である。図 2.6 に、$N=8$のときの変換係数$F(u,v)$に対する基底パターンを示す。同図左上の成分$F(0,0)$が直流（DC：Direct Current）成分とよばれ、右下の成分ほど高周波となっている。[10]

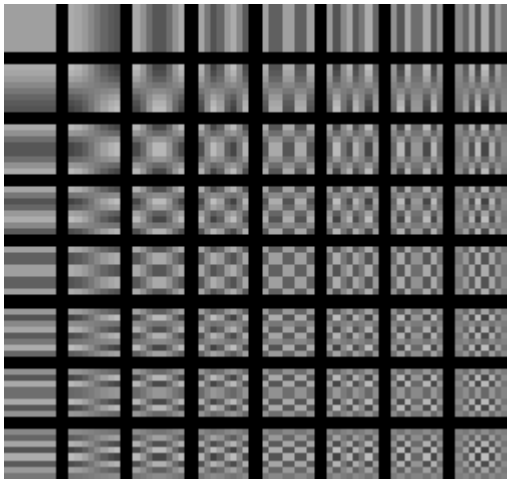

図 2.6　8×8 ブロックに対する DCT の基底パターン[11]

2.2.5 エントロピー符号化

エントロピー符号化は、出現頻度の高いシンボルには短い符号語を、出現頻度の低いシンボルには長い符号語を割り当てる符号化である。このことで情報の理論的圧縮限界であるエントロピー（平均情報量）に近い符号量で符号化することが可能となる。この方式では符号長が可変となるため可変長符号化とも呼ばれる。

前節で示した DCT の後に量子化を行うと高周波成分が 0 となることが多い。そこで、低周波成分から高周波成分にかけてジグザグにスキャンして値を並べると、後ろのほうが 0 の連続して現れるデータ列になる。そこで「0,0,0,0,0,0」といった 0 の連続を「0 が 6 つ」というようなコンパクトな表現にすることができる。このように、0 の連続する個数（ゼロラン長）とその直後の 0 でない係数（レベル）のペアで表現する符号化をラン-レベル符号化という。

AVC や HEVC では、CABAC（Context-based Adaptive Binary Arithmetic Coding）で符号化が行われる。図 2.7 に、CABAC の処理の概要を示す。まず、符号化対象のデータを 2 値信号に変換する。その後あらかじめ用意された複数の確率モデルの中から、符号化済みのデータ系列に従い使用するコンテキストモデルを選択する。そして最後に算術符号化を行う。算術符号化とは、0 以上 1 未満の数直線をシンボルの出現確率に応じて Augend と呼ばれる領域に分割して、シンボル系列に対応する Augend を最小精度の 2 新数で表現する方法である。図 2.8 に算術符号化の原理を示す。また、算術符号化はシンボルの出現確率が均一に近いデータでは圧縮が見込めないため、処理の高速化を図るバイパスモードも用意されている。 [12]

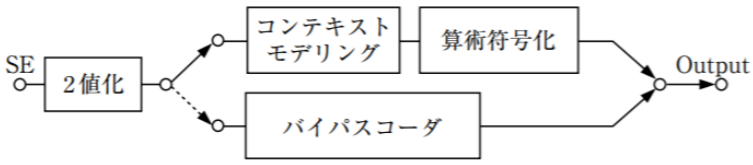

図 2.7　CABAC の処理の概要[12]

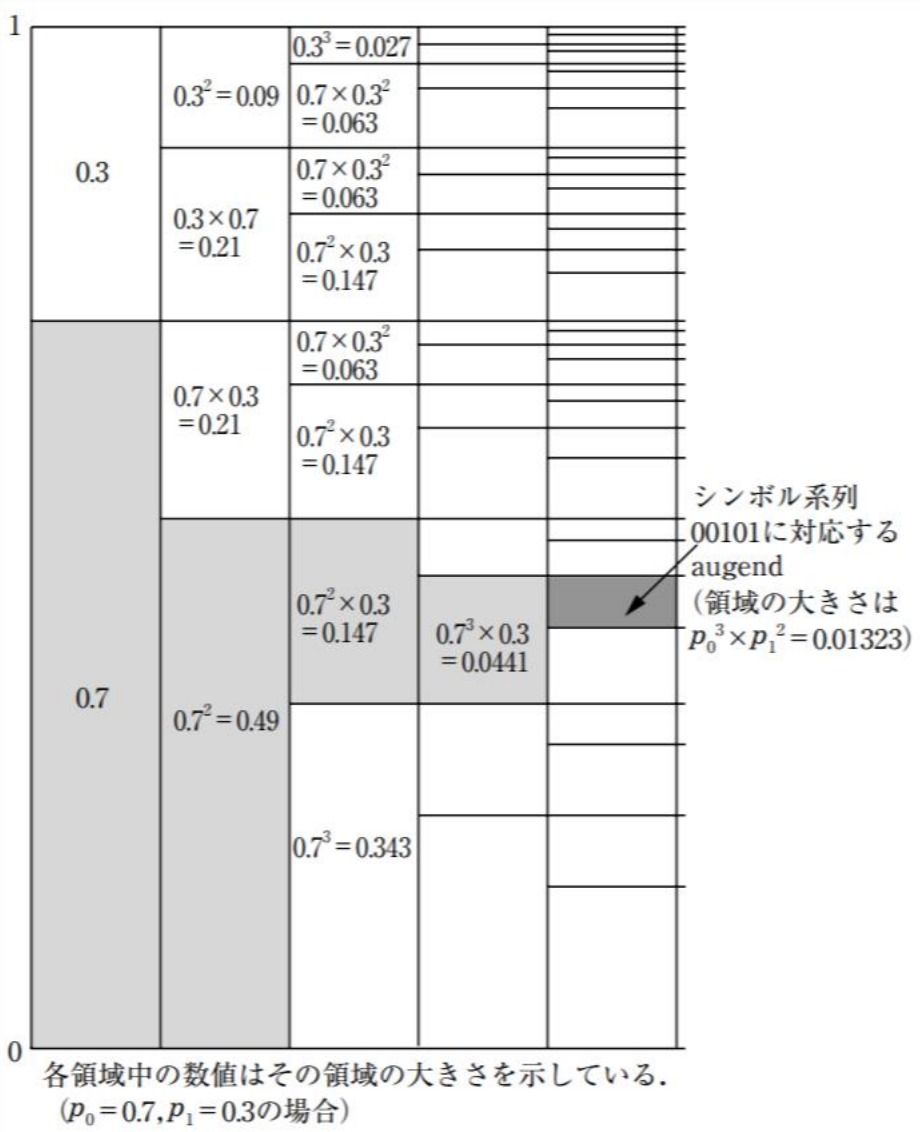

図 2.8　算術符号化の原理[12]

2.3 動画像符号化標準の歴史

動画像符号化に関する国際標準は、ITU-T（ITU - Telecommunication Standardization Sector）と ISO/IEC JTC1（International Organization for Standardization / International Electrotechnical Commission Joint Technical Committee）の 2 つの国際標準化機関によって進められてきた。ITU-T は VCEG（Video Coding Experts Group）によって H シリーズを、ISO/IEC JTC1 は MPEG（Moving Picture Experts Group）によって MPEG シリーズを成立させている。図 2.9 に、動画像符号化標準制定の経過を示す。

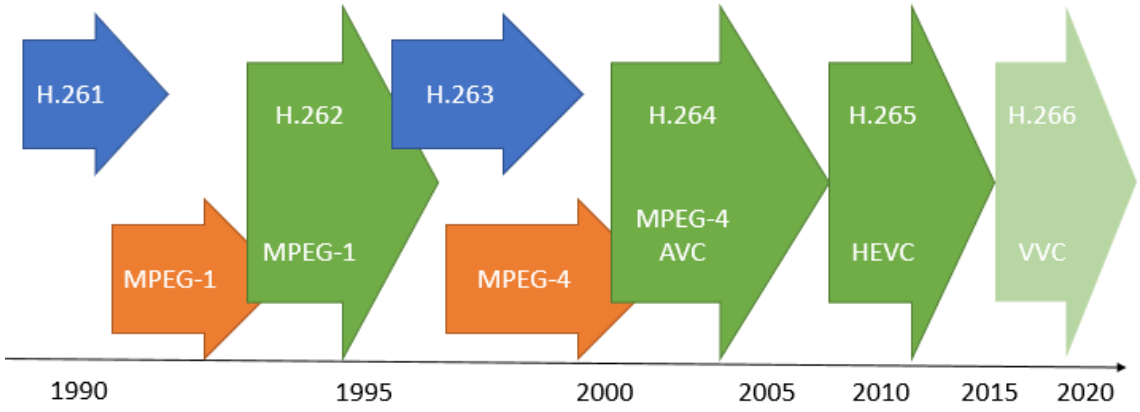

図 2.9　ITU-T と MPEG による動画像符号化標準制定

ITU-T （当時 CCITT、Consultative Committee of International Telegraph and Telephone）の H.261 が標準化された 1990 年代以降は、ハイブリッド符号化が基本となっている。ハイブリッド符号化とは、前節で述べた前節で述べた画面内予測・画面間動き補償予測といった「予測」と DCT などの「変換」を組み合わせて行う方式のことである。

H.261 はそれ以降の符号化標準の原型となるものだったが、通信での用途を想定したため CD-ROM などの DSM（Digital Storage Media）への蓄積を想定していた MPEG とは時間やコストといった面で条件が異なるものだった。そのような条件の違いに基づき、MPEG-1 では過去と未来の双方向から行う予測や動きベクトルの精度を整数画素から 1/2 画素の単位への向上が採用された。こうした双方向予測を用い、周期的に画面内予測を挟むことにより、符号化効率を向上させるとともに、デバイスで要求される特殊再生やランダムアクセスを可能とした。

MPEG-1 に次いで、MPEG-2 の標準化活動が始まった。5～10Mbps でアナログ標準テレビジョン（SDTV）の解像度を実現することが目標として定められた。そして DSM に限らず多様なアプリケーションで用いられる汎用的な符号化を目指す方針で、ITU-T と合同で標準化作業が行われた。ITU-T 側では H.262 と番号づけられたが内容は MPEG-2 と全く同一である。MPEG-2 は、インターレース（飛び越し走査）信号へ対応しているという点で MPEG-1 と大きく異なる。MPEG-2 では効率よくインターレース信号を符号化するため、フィールド/フレーム適応の予測や DCT を導入している。また、階層符号化によるスケーラビリティ機能や、プロファイルおよびレベルによって使用する符号化ツールを定義する機能によって、様々な用途やデバイス性能、解像度などに対応することが可能となった。標準化が進むにつれ、当初の目標だった標準テレビジョンだけでなく HDTV も包含することとなった。その結果、現在デジタルテレビ放送や DVD などで広く用いられる規格となった。

H.262/MPEG-2 に続いて、ITU-T 側では H.263 の標準化が行われた。H.263 はアナログ電話網において低ビットレートでのテレビ電話を実現するために標準化が開始された。H.261 を基に、ループフィルタを除き 1/2 画素制度動き補償を採用、シンタックスのオーバーヘッドを削減、DCT 係数の可変長符号化においてラン-レベルにブロック内の最後のレベルであるかというフラグを追加し 3 次元化という変更が加えられた。その結果特に低レートで符号化効率が大きく向上された。一方、MPEG 側では MPEG-4 の標準化が行われた。MPEG-4 では第 3 世代携帯電話のような低速低品質の回線の使用環境や低解像度映像への対応が重視された。そのため動きベクトルを 1/4 画素精度にし、予測単位のブロックサイズを 16×16 だけでなく 8×8 も使用可能とするなどにより動き補償予測が強化されたほか、エントロピー符号化での誤り耐性も強化された。さらに、映像を背景や動く人物といったオブジェクトに分割して符号化する方式も定義されている。

次いで標準化されたのが VCEG と MPEG が JVET（Joint Video Team）を設立し合同で開発した H.264/AVC（MPEG-4 AVC）である。AVC では予測ブロックサイズを 7 種類に増やし、参照できるフレームの数も増やした。画面内予測については、DCT の前に複数種類の方向から選択して行う手法が追加された。DCT については 8×8 から 4×4 にして歪みを目立ちにくくし、整数で行うよう変更された（後に大画面を扱える 8×8DCT も併用可とされた）。エントロピー符号化では符号化済みの結果をコンテキストとして適応的に行う算術符号化が導入された。さらに、符号化によるゆがみを低減させるデブロッキングフィルタが追加された。こうした変更により、従来の 2 倍の圧縮効率が実現された。そのような圧縮効率の高さから、テレビ会議システムや携帯電話、ワンセグ、IPTV、Blu-ray など様々なアプリケーションに用いられている。[10]

H.264/AVC の後、H.265/HEVC の標準化が行われた。HEVC でも AVC と同様に JCT-VC という合同グループが作られ標準化作業が進められた。AVC と比較して、ブロック分割方法やブロックサイズのバリエーション、エントロピー符号化の方法やループ内フィルタなどが異なる。画面内予測については予測モードが 9 種類から 35 種類まで大幅に増やされた。詳細については 3 章で述べる。このような要素技術の改善により AVC の 2 倍の圧縮効率が実現された。[13]

H.265/HEVC の次の規格として、現在 H.266/VVC（Versatile Video Coding）の標準化が進められている。VVC も HEVC と同様で JVET（Joint Video Exploration Team）という共同チームで標準化が行われている。2020 年の最終国際規格案（FDIS：Final Draft International Standard）発行に向けて作業が進められている[14]。符号化効率は、HEVC と比べて 30%以上の向上が見込まれている[15]。

2.4 画質評価

ここでは画質評価に用いられる尺度について説明する。画質評価には主観評価と客観評価がある。主観評価は人間の感覚に基づき評価値を決定する方法であるのに対し、客観評価は信号解析により定量的に評価を行うため比較的容易である。以下では、客観評価としてよく用いられる PSNR（Peak Signal to Noise Ratio）と SSIM（Structural SIMilarilty）について説明する。

2.4.1 PSNR

PSNR は式(2.8)のように定義される。

$$PSNR = 10\log_{10}\frac{S^2}{MSE} \tag{2.8}$$

Sは信号のピーク間の変位を示す。8bit の画像の場合、信号は 0～255 の範囲となるため、$S = 255$となる。MSEは原画像と復号画像の平均 2 乗誤差（Mean Squared Error）を指し、$x(i)$を原画像のi番目の画素位置の信号、$y(i)$を複合画像のi番目の画素位置の信号、Nを画像に含まれる信号の総数として式(2.9)のように表される。

$$MSE = \frac{1}{N}\sum_{i=0}^{N-1}[x(i) - y(i)]^2 \tag{2.9}$$

PSNR の特徴として劣化度合いの大きい画像や画質の偏りのある画像では主観画質と異なることがあるとされる一方、複雑な計算を要さないため HEVC をはじめとして広く用いられている。

HEVC などでは PSNR と bitrate から RD 曲線を描き、それらを 3 次関数で近似して bitrate 軸方向に積分した値とベースラインの同様に算出した値の差をとった BD-Rate（Bjontegaard Delta-Rate）[16]という指標が符号化性能評価に用いられている。BD-Rate は同一 PSNR の時のビットレートの差を表し、負の値であればベースラインよりも符号化性能が高いことになる。

2.4.2 SSIM

SSIM は、Wang らによって提案された指標で、画像構造の類似度が人間の画質劣化の知覚に寄与するという仮説に基づいて定義されている。原画像を$x(i)$、復号画像を$y(i)$とすると SSIM は式(2.10)のように表せる。

$$SSIM(x,y) = [l(x,y)]^{\alpha}[c(x,y)]^{\beta}[s(x,y)]^{\gamma} \tag{2.10}$$

ここで、α、β、γは正の定数で、Wang らはすべて 1 としている[17]。また、$l(x,y)$、$c(x,y)$、$s(x,y)$はそれぞれ原画像と復号画像の輝度の比較項、コントラストの比較項、画像構造の比較項を表しており、式(2.11)～(2.13)に示す。

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{{\mu_x}^2 + {\mu_y}^2 + C_1} \tag{2.11}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{{\sigma_x}^2 + {\sigma_y}^2 + C_2} \tag{2.12}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{2.13}$$

Sを信号のピーク間の変位として$C_1 = (K_1S)^2$、$C_2 = (K_2S)^2$で、$C_3 = C_2/2$とされ$K_1 = 0.01$、$K_2 = 0.03$が良いとされる。μ_x、σ_x、σ_{xy}はそれぞれ 11×11 の正規化されたガウス関数で重みづけられた$x(i)$の平均、$x(i)$の標準偏差、$x(i)$と$y(i)$の共分散である。以上をまとめると式(2.14)のようになる。

$$SSIM(x,y) = \frac{(2\mu_x\mu_n + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{2.14}$$

式(2.14)は有限領域の重みづけが適用されているため画面中の小領域ごとに定義されることとなる。画面全体の評価はこの各小領域の評価値の平均により与えられ MSSIM（Mean SSIM）という。[18]

2.5 Neural Network

Neural Network は、いくつかのノード（信号）をそれぞれ重みづけて別のノードに入力し足し合わすなどの処理を行う、といった変換がいくつも連なってできるものである。図 2.10 に Neural Network の例を示す。図中の○がノードを表している。

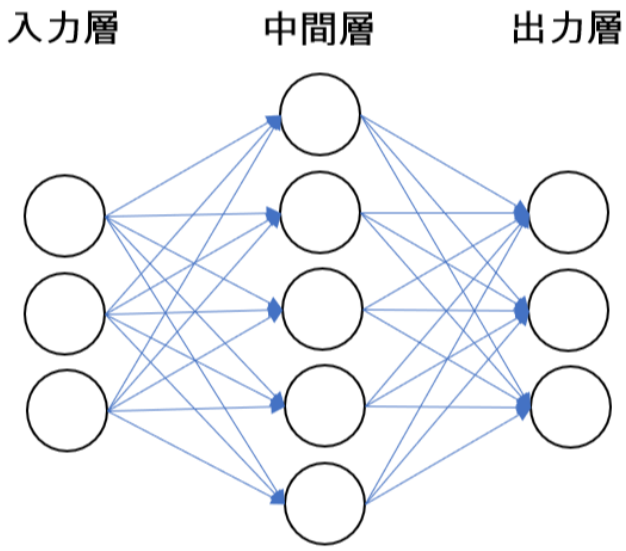

図 2.10　Neural Network の例

2000 年代、画像認識分野では SIFT などの特徴量と SVM（Support Vector Machine）などの識別器を組み合わせた方法が主流だった。しかし、2012 年の ILSVRC（ImageNet Large Scale Visual Recognition Competition）という画像認識のコンペティションで AlexNet という CNN（Convolutional Neural Network）を用いた手法が他に大きく差をつけて優勝したのをきっかけに Neural Network が注目されるようになってきた。画像認識の他にも[19][20]など様々なタスクで利用されている。[21]

以下では、CNN や代表的な CNN の構造、活性化関数、損失関数、最適化手法について述べる。

2.5.1 CNN

CNN は、中間層に Convolution（畳み込み）層を用いる Neural Network である。先に示した図 2.10 のようなネットワークでは 1 次元のデータを扱うため、画像データを入

力するには 1 次元化しなければならず空間情報が失われる。CNN はそのようなネットワークとは異なり空間情報を活かすネットワークであり、画像処理でよく用いられている。

Convolution 層では畳み込み演算が行われるが、これは画像のフィルタリング処理と同様の処理である。畳み込み演算のパラメータとして Kernel Size、Channel 数、Stride、Padding がある。Kernel size は、畳み込みのフィルタ（Kernel）の大きさのことである。Channel 数は、特徴を取り出すために何種類のフィルタを用いるか定める数である。Stride は、フィルタをスライドするときの間隔である。Padding は、畳み込みの前に入力データの周りにデータを埋める処理およびその際どれだけの幅で行うか定める数である。図 2.11 に畳み込み演算の例を示す。同図では、Kernel Size 3×3、Channel 数 1、Stride 2、Padding 1 で畳み込みが行われている。

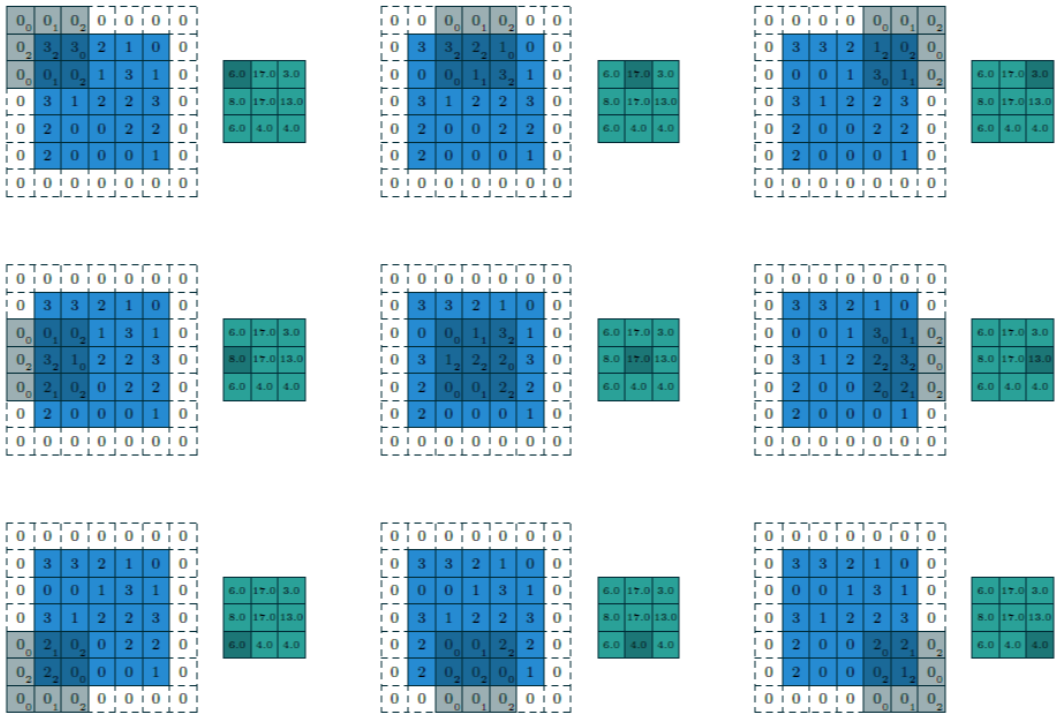

図 2.11　畳み込み演算の例[22]

畳み込み演算の後には、しばしば Pooling という処理が行われる。Pooling は畳み込みのようにフィルタを用いる演算だが、畳み込みとは異なりフィルタの係数は学習されず決まった処理を行う。よく使われるものとしてフィルタ内の最大値をとる Max Pooling や平均値をとる Average Pooling が挙げられる。

2.5.2 代表的な CNN の構造

LeNet は 1990 年代に提案されていたネットワークだがハードウェアの制約から 2010 年まで実装が困難だった。しかし、back propagation（誤差逆伝播）を用いた学習の結果、手書き文字認識で高い性能を発揮した。LeNet は 2 つの Convolution 層の後に 2 つの全結合（FC：Fully Connected）層があり、各 Convolution 層の後にサブサンプリングが行われるような構造になっている。図 2.12 に LeNet の構造を示す。[23]

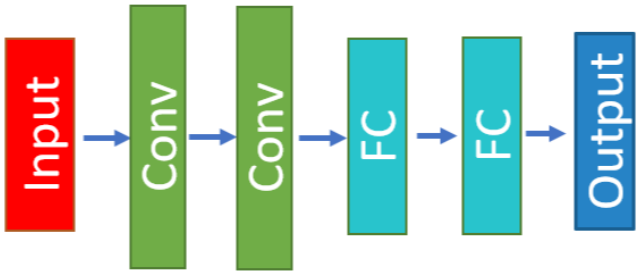

図 2.12　LeNet の構造

AlexNet は、LeNet よりも多層（deep）で Channel 数も多い（wide な）ネットワークである。AlexNet は 5 つの Convolution 層の後に 3 つの FC 層がある構造になっている。1 層目と 2 層目は畳み込みの後に Max Pooling（MP）が行われる。Local Response Normalization（LRN）というチャンネル方向の正規化や Dropout という一定確率でのニューロンの無効化といった新しい技術も導入された。図 2.13 に AlexNet の構造を示す。[24]

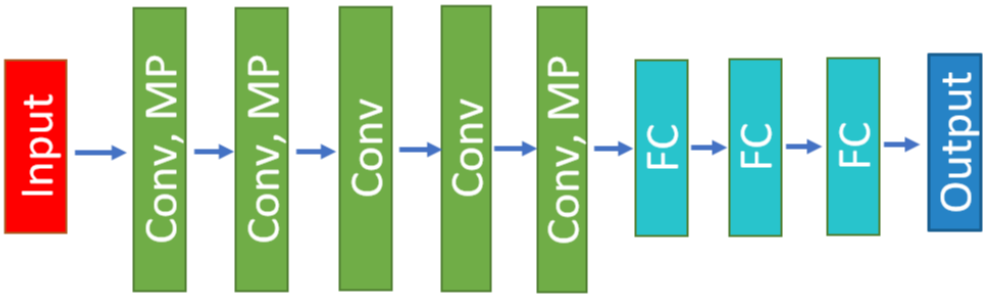

図 2.13　AlexNet の構造

AlexNet 以降、VGG[25]、ResNet[26]といった、より deep なネットワークも出現していて、画像認識などのタスクで精度向上を実現している。一方でその分、ハードウェアのコストも増大している。[27]

2.5.3 活性化関数

活性化関数とは、畳み込みなど層での演算の後に適用される関数のことである。以下では、代表的な活性化関数として、ReLU（Rectified Linear Unit）、Sigmoid、Tanh について説明する。ReLU は、0 以上の数はそのまま、0 未満の数は 0 とするような関数である。Sigmoid は式(2.15)で、Tanh は式(2.16)で表される関数である。

$$Sigmoid(x) = \frac{1}{(1 + e^{-x})} \tag{2.15}$$

$$Tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.16}$$

図 2.14 にそれぞれの活性化関数の概形を示す。同図より、Sigmoid と Tanh は値域が異なるものの、滑らかな曲線という点で似ていることが確認できる。

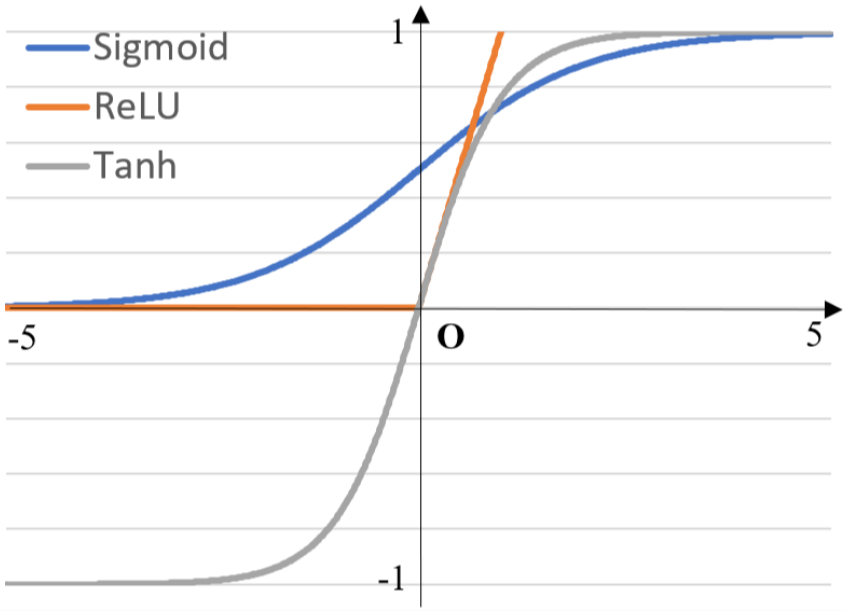

図 2.14 活性化関数の概形

2.5.4 損失関数

損失関数は、Neural Network の学習に用いられる指標である。損失関数により得られた値、すなわち損失（loss）が大きいほど本来得るべき値との誤差が大きく性能が悪いことが示される。

よく用いられる損失関数として交差エントロピー誤差が挙げられる。交差エントロピー誤差は、Neural Network の出力を$y(i)$、対応する正解ラベルを$t(i)$として式(2.17)のように表される。

$$E = -\sum_i t(i) \log[y(i)] \tag{2.17}$$

また、損失関数の前で Neural Network の出力に Softmax 関数を適用することもしばしばある。Softmax 関数は式(2.18)のように表されるもので、Neural Network の出力を 0 から 1 の確率のような意味を持たせた値にする効果がある。

$$Softmax = \frac{\exp[a(i)]}{\sum_i \exp[a(i)]} \tag{2.18}$$

2.5.5 最適化手法

最適化手法は、loss を小さくするために最適なパラメータを求めるのに用いられる方法である。

最も単純なのが勾配を用いる方法で、パラメータに対する損失関数の勾配を求めて損失関数の低い方向へパラメータを動かすことを繰り返し行うものである。これを勾配降下法という。一方、最適化の際に training データからランダムに複数のデータを取り出してミニバッチという単位として、それら毎に loss を減らすようにパラメータを更新する学習をミニバッチ学習という。このミニバッチ学習を適用して行う勾配降下法を確率的勾配降下法（SGD：Stochastic Gradient Descent）という。

よく用いられる手法として Adam[28]が挙げられる。Adam は、勾配方向に力を受けて加速するという物理法則に準じるような Momentum と学習が進むにつれて学習係数（パラメータを動かす大きさを決める係数）を小さくする AdaGrad[29]の 2 つの最適化手法をかけあわせたような手法となっている。[30]

第 3 章 HM における画面内予測

本章では、本研究と密接に関わる HM における画面内予測について説明する。はじめに、予測を行うブロックへの分割について説明し、その後画面内予測モードの種類と決定方法について説明する。

3.1 ブロック分割

HEVC ではピクチャをブロックに分割して符号化処理を行う。まず CTU（Coding Tree Unit）という符号化の基本単位となる 64×64 のブロックに分割する。そして CTU は CU（Coding Unit）という可変サイズのブロックに再帰的に分割する。CU のサイズとして 8×8、16×16、32×32、64×64 が取り得る。そして CU は PU（Prediction Unit）という予測単位のブロックに分割される。画面内予測の場合、8×8 の CU のみ 4×4 の PU に分割するかどうか選択でき、それ以外は CU がそのまま PU となる。図 3.1 に、ブロック分割と PU サイズのバリエーションを示す。

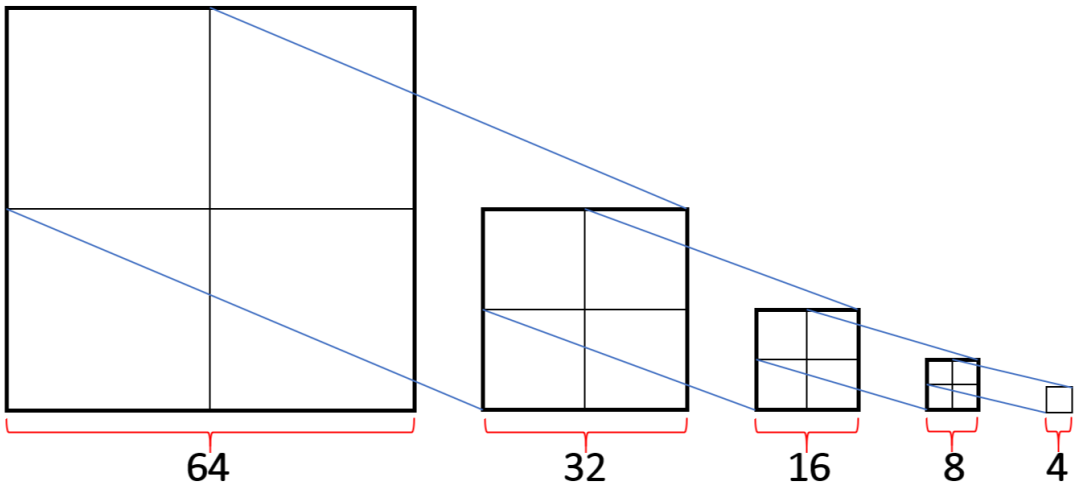

図 3.1　ブロック分割と PU サイズのバリエーション

3.2 画面内予測モードの種類

HEVC の画面内予測モードとして、Planar、DC、33 種類の Angular の計 35 種類が用意されている。本節では、それぞれのモードでの画面内予測処理について説明する。[13]

3.2.1 Planar 予測

Planar 予測は、4 つの参照画素値を用いて滑らかに予測画素を生成するモードである。ブロックサイズN、位置(x, y)の場合、$(-1, N)$、$(-1, y)$、$(x, -1)$、$(N, -1)$の位置の画素値を参照し、$(-1, N)$が$(x, N-1)$に、$(N, -1)$が$(N-1, y)$にあると見立てて線形補間して予測値を導出する。図 3.2 に、Planar 予測を可視化した図を示す。

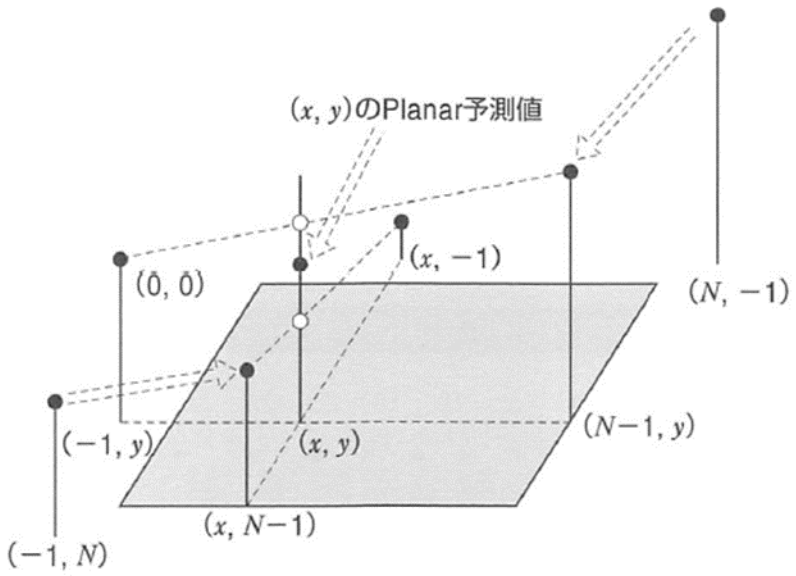

図 3.2　Planar 予測の可視化[13]

3.2.2 DC 予測

DC 予測は、参照画素の平均値で予測領域を埋めるモードである。ブロックサイズをNとして、ブロックの左隣のN個と上隣のN個の計 $2N$個の画素を参照する。

3.2.3 Angular 予測

Angular（方向性）予測は、33 種類の用意された方向のうち 1 つを選択し、その方向の参照画素から予測画素を生成するモードである。図 3.3 に方向性予測の参照方向を示す。同図より、自然画像に多く含まれる水平・垂直のエッジを考慮して水平・垂直付近は細かくモードが設定されていることが確認できる。

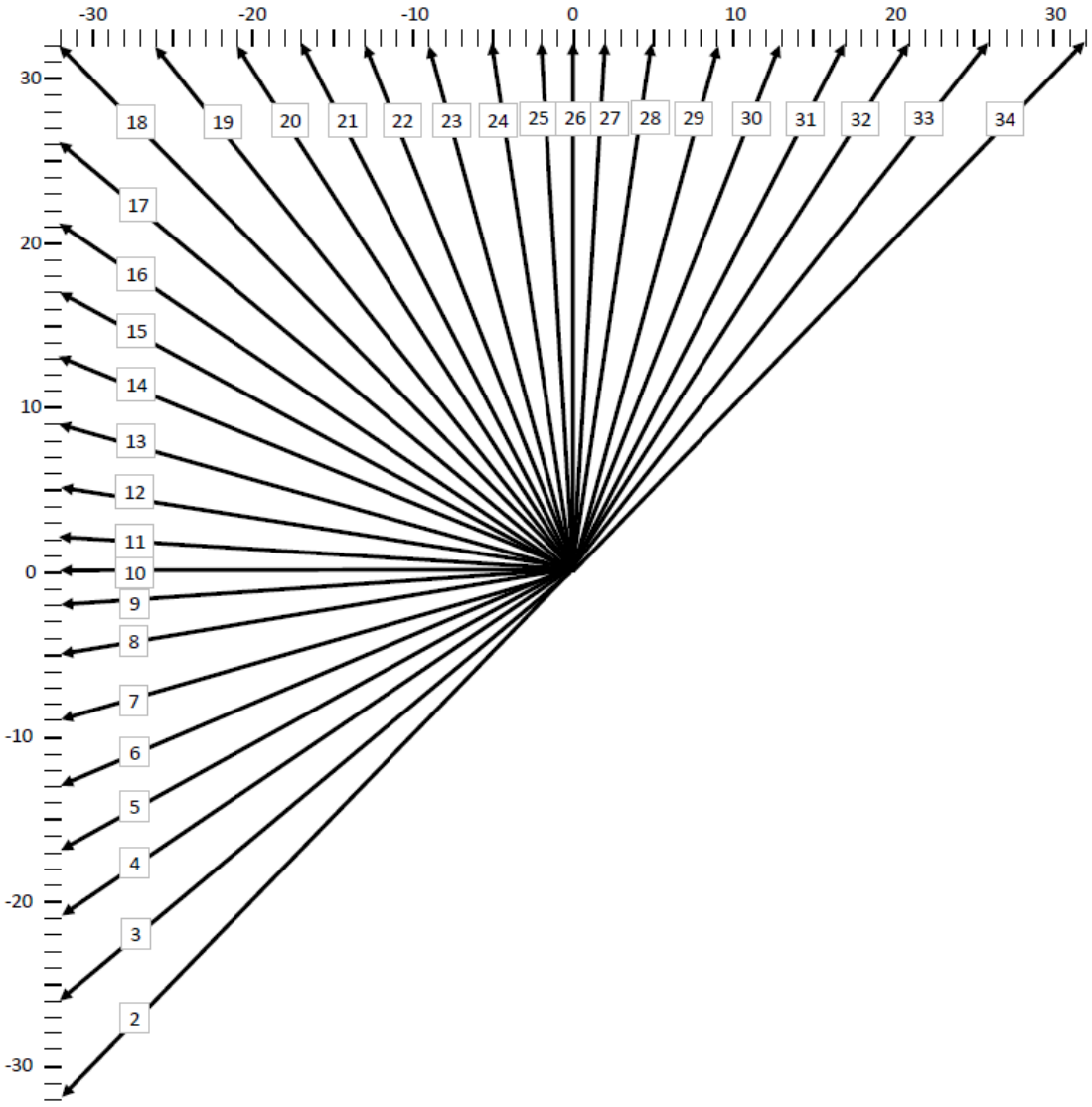

図 3.3　方向性予測の参照方向

3.3 画面内予測モードの決定

本節では、画面内予測モードを決定するために用いられるコスト関数、MPM（Most Probable Mode）と画面内予測モード決定の流れについて説明する。

3.3.1 コスト関数

画面内予測モードは、RD 最適となるように選択される。HM においては RD コストが最小となるようなモードが選択される。RD コストは式(3.1)で定義される。

$$J = D + \lambda R \tag{3.1}$$

この式において、Rが符号量、Dが符号化による歪みを表す。そして、λは定数で大きくするほど Rate を重視する符号化、すなわち低ビットレートでの最適化を可能にする。量子化ステップが小さいほど符号化による歪みが小さくなるため、λは量子化ステップを定める QP（Quantization Parameter）の関数で設定される。符号化による歪みDは、符号化対象画像と復号画像の 2 乗誤差和で表される。

しかし、符号化対象画像と復号画像の 2 乗誤差和を算出するためには、計算コストの高い変換・乗算処理などが必要となる。そのため、符号化による歪みDを簡単にしたコスト関数が近似として用いられる。SATD（Sum of Absolute Transform Difference）は、符号化対象画像と予測画像の誤差に対して簡単な直交変換であるアダマール変換を行い、絶対値誤差和をとったものである。これを用いたコスト関数が SATD コストで、式(3.2)で表される。

$$J = SATD + \lambda R \tag{3.2}$$

3.3.2 MPM

MPM（Most Probable Mode）は、左と上の隣接する PU で用いられたモードから最適な画面内予測モードを推定して得る 3 つのモードである。これは、隣接するブロックはモードにある程度相関があるということを利用している。また、35 種類の画面内予測モードから 1 つ選択するより 3 つの MPM から 1 つ選択するほうが、選択されたモード番号を符号化するときに必要なシンボル数が少なく済むため、MPM には符号量を減らす役割も果たしている。[31]

図 3.4 に、MPM 取得の流れを示す。左と上の隣接する PU で用いられたモードが同じで Angular モードであればそのモードを 0 番目に、左隣のモードを 1 番目に、右隣のモー

ドを 2 番目に設定する。ただし、隣のモードがない場合は 180 度回転させたときのモードの隣のモードを設定する。左と上の隣接する PU で用いられたモードが同じで Angular モードでなければ Planar を 0 番目に、DC を 1 番目に、Vertical を 2 番目に設定する。左と上の隣接する PU で用いられたモードが異なる場合は、左隣の PU で用いられたモードを 0 番目に、上隣の PU で用いられたモードを 1 番目にする。その後、Planar、DC、Vertical の優先順で、左と上の隣接する PU で用いられてないモードを 2 番目にする。

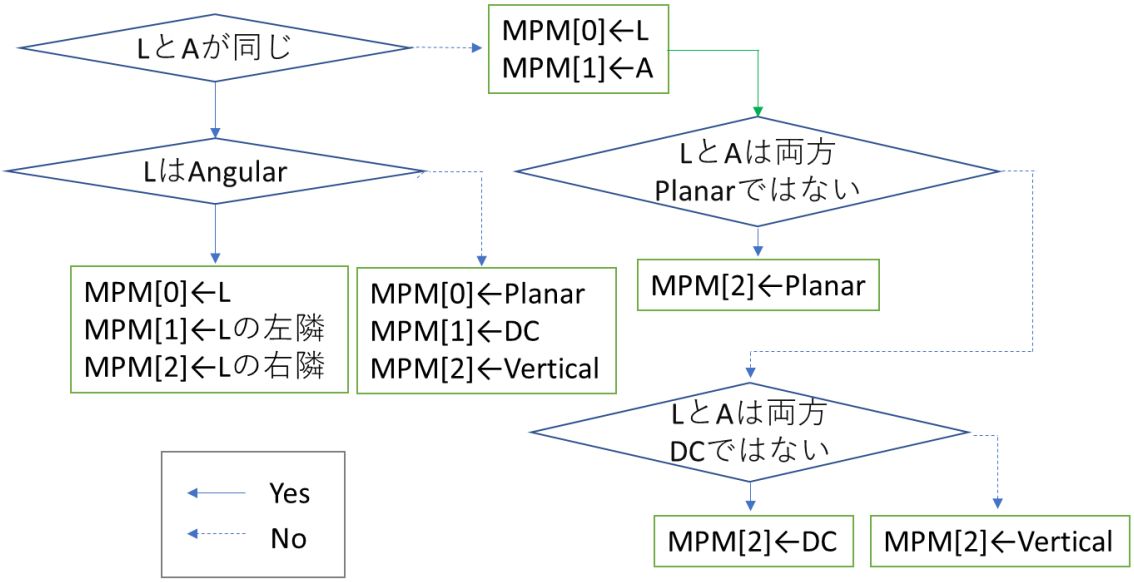

図 3.4　MPM 取得のフロー

3.3.3 画面内予測モードの決定の流れ

画面内予測モードは、先に述べたコスト関数と MPM を用いて決定される。図 3.5 に画面内予測モード決定の流れを示す。まず、比較的計算量の少ない MPM と SATD コストを用いて 35 種類のモードを絞り込む。MPM については、左と上の隣接する PU で用いられたモードが同じであれば 1 つ、そうでなければ 2 つが RD コスト計算対象モードとされる。SATD コストは 35 種類のモードすべてについて計算されてコストの低い順に、PU のサイズが 4×4 か 8×8 の場合は 8 個、そうでなければ 3 個が RD コスト計算対象モードとされる。そして、絞り込まれたモードについて RD コストを計算し最もコストの低いモードを選択する。

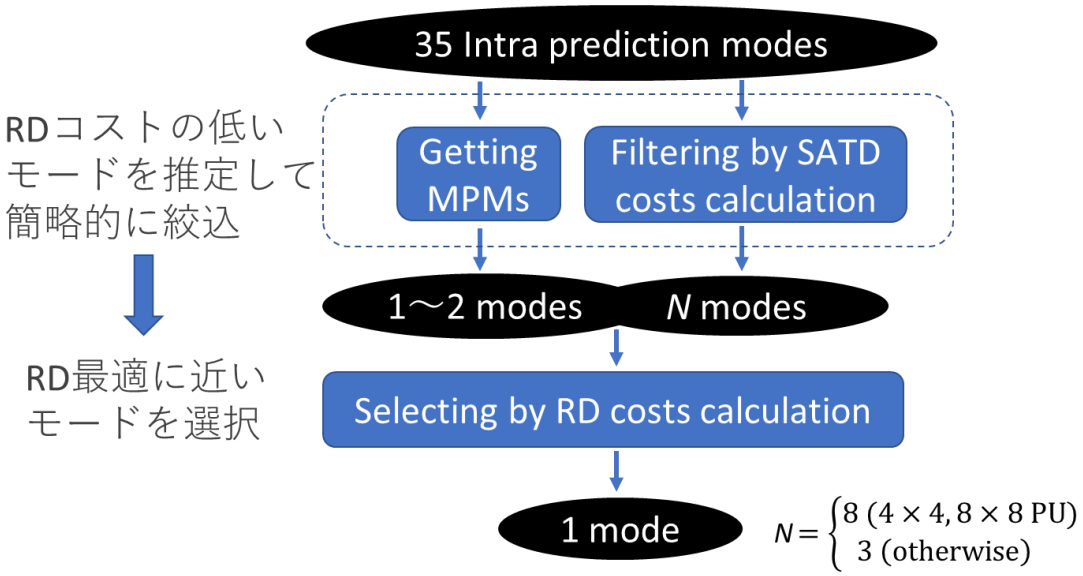

図 3.5　画面内予測モード決定の流れ

第 4 章 CNN による最適画面内予測モード推定

本章では、第 3 章で述べた HM において画面内予測モードを決定する箇所に、CNN を用いてより最適なモード推定を行う先行研究の CNNMC と、それを実装して推定精度を評価した実験について述べる。

4.1 CNNMC による最適画面内予測モード推定

CNN を用いた最適画面内予測モードの確率分布を求める手法として CNNMC（CNN Mode Coding）[32]がある。本節では、CNNMC のネットワーク構造、学習・評価用のデータセットについて説明する。

4.1.1 CNNMC のネットワーク構造

CNNMC の概要を図 4.1 に示し、CNNMC で用いられる CNN の構造とパラメータを、それぞれ図 4.2、表 4.1 に示す。

ネットワークの入力は、予測対象ブロックの左・左上・上に隣接する 3 つの 128×128 の復号済みブロックの輝度画素値（Reconstructed Pixels）と、それらに対応する 4×4 のブロック毎の最適画面内予測モードが格納された 32×32 の 3 つの行列（Intra Prediction Modes）である。各入力は学習しやすいよう標準化、正規化される。[32]には標準化、正規化についての詳細な記述がないため、本研究では最大値を 1、最小値を 0 とする正規化と解釈している。ネットワークの出力は、4×4 のブロック毎の最適画面内予測モードの確率分布（Probability Distribution）である。4N×4N（N＝2、4、8、16）のブロック毎の確率分布は、ネットワークの出力に対し N×N の Kernel を用いて Average Pooling を行うことで得られる。Reconstructed Pixels の入力は 5 つの Convolution 層を、Intra Prediction Modes の入力は 1 つの Convolution 層を経て足し合わされた後 Softmax 関数にかけられる。各 Convolution 層の後には活性化関数として Tanh が用いられる。

CNNMC は画面内予測モードを 67 に増やした場合を想定したネットワークになっているためネットワークの出力が 67 チャンネルになるが、HM の 35 種類の画面内予測モードを対象とするデータセットで学習を行えば余剰なチャンネルが学習されないので無視でき 35 チャンネルの出力とみなせる。そのため本研究では、出力のチャンネル数を変更せずこのネットワークを扱う。

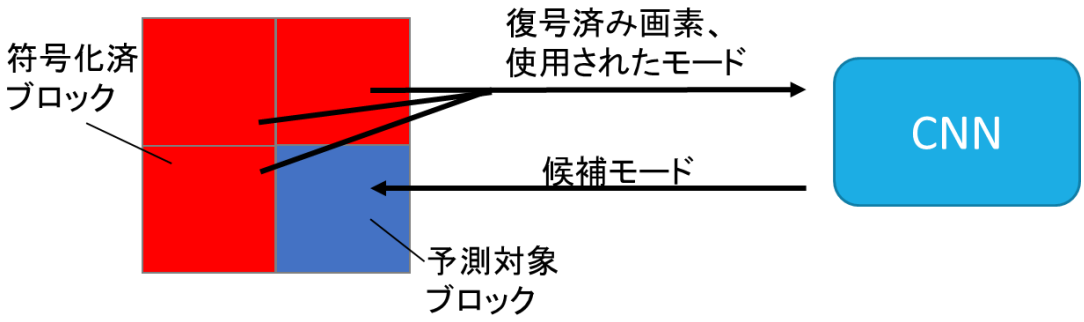

図 4.1　CNNMC の概要

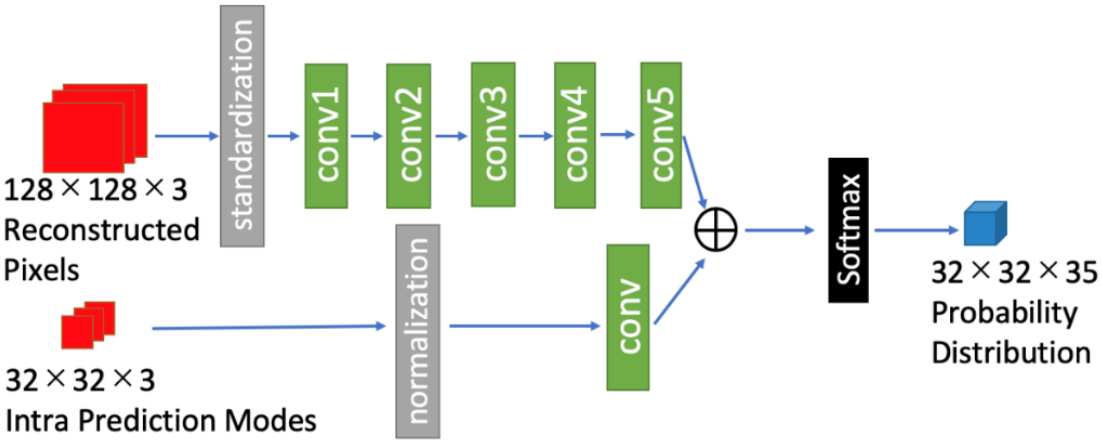

図 4.2　CNNMC で用いられる CNN の構造

表 4.1　CNNMC で用いられる CNN のパラメータ

Layer	Kernel Size	Channel 数	Stride
conv1	4×4	128	1
conv2	4×4	128	2
conv3	3×3	64	1
conv4	4×4	64	2
conv5	1×1	67	1
conv	1×1	67	1

4.1.2 CNNMC の学習・評価用のデータセット

本節では、前節で述べたような最適画面内予測モード推定を行うためのデータセットについて説明する。

データセットの作成にあたって、[32]と同様に CVPR 2018 の Workshop and Challenge on Learned Image Compression（CLIC）[33] Dataset P（“professional”）と Dataset M（“mobile”）の画像を使用する。画像は training 用が 1633 枚、validation 用が 102 枚となっている。training 用の画像から CNN の学習用のデータセットを、validation 用の画像から評価テスト用のデータセットを作成する。それぞれの画像は、HM によりエンコードしてその際に各予測ブロックに対応する Reconstructed Pixels、Intra Prediction Modes、Target Modes の情報を取り出して組にすることでデータセットを作成する。エンコードは encoder_intra_main.cfg の設定ファイルを用いて All Intra で行う。さらに、MPM がシンボル数で有利であるために多く選択されることを防ぐため、MPM を使わないようにし、35 種類全ての画面内予測モードの RD コストを計算（Full RD Search）する。そして RD コストが最小のモードを Target Mode とする。これにより、RD コストが最小となる画面内予測モードを推定するネットワークとなるよう学習させることができる。図 4.3 にデータセット作成の概要を示す。

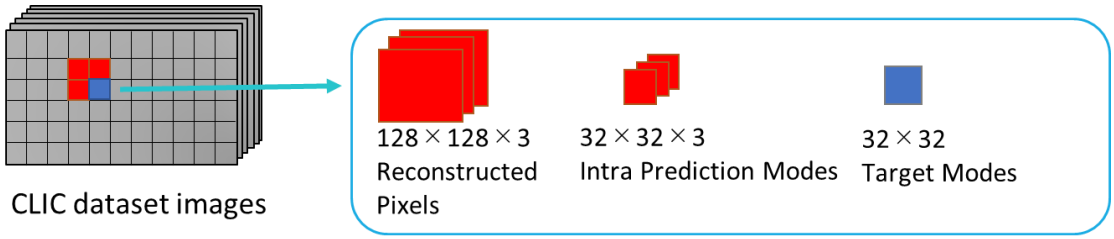

図 4.3　データセット作成の概要

4.2 CNNMC のネットワークを用いた最適画面内予測モード推定の精度評価実験

本節では、前節で示したデータセットで学習させた CNNMC のネットワークを用いて最適画面内予測モード推定を行った精度評価結果を示す。その際、ネットワークのパラメータを変更した異なるネットワーク構造を用いた場合と、MPM と様々に組み合わせて推定を行った場合の結果の比較もそれぞれ行う。図 4.4 に精度評価の概要を示す。精度評価は、テスト用データセットの Reconstructed Pixels と Intra Prediction Modes を学習済み

CNN に入力し得られた候補モードと HM から得た MPM の組み合わせの中に Target Mode が含まれている割合を算出することで行う。

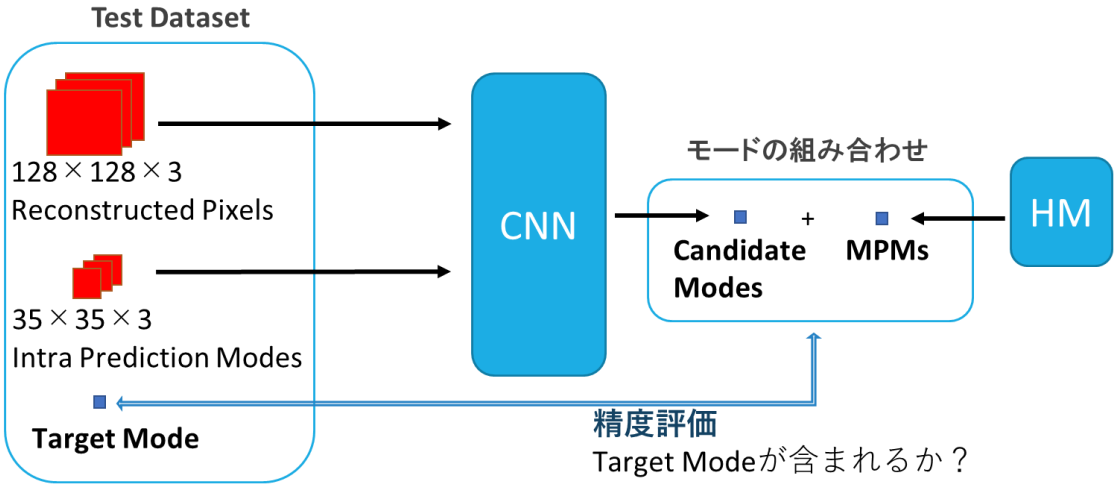

図 4.4　精度評価の概要

4.2.1 比較するネットワーク構造とモードの組み合わせ

異なるネットワーク構造の比較として、conv1～conv4 層の Kernel Size と Channel 数を変更したものを検討する。表 4.2 に比較したネットワーク構造を示す。例として、構造(2-3)は、Channel 数が表 4.1 の 2 倍で Kernel Size が全て 3×3 のネットワーク構造を指す。

表 4.2　比較したネットワーク構造

		Channel 数		
		表 4.1 と同じ	表 4.1 の 1/2 倍	表 4.1 の 2 倍
Kernel Size	表 4.1 と同じ	(1-1)	(1-2)	(1-3)
	全て 3×3	(2-1)	(2-2)	(2-3)
	全て 5×5	(3-1)	(3-2)	(3-3)

CNN からの出力モードと MPM のモードの組み合わせとしては、表 4.3 のものを検討する。組み合わせ(a)は CNN から出力された 3 つのモードのみで推定を行い、組み合わせ(i)は MPM の 3 つのモードのみで推定を行うことを表している。

表 4.3　比較した CNN からの出力モードと MPM のモードの組み合わせ

	CNN からの出力モード数	MPM のモード数
(a)	3	0
(b)	1	1
(c)	2	1
(d)	1	2
(e)	2	2
(f)	3	1
(g)	3	2
(h)	3	3
(i)	0	3

4.2.2 実験環境

実験に使用した計算環境を表 4.4 に示す。

表 4.4　実験に使用した計算環境

マシン 1	CPU	Intel Core i7-8700K @ 3.70GHz ×12
	メモリ	15.6GiB
	GPU	GeForce GTX 1080 8111MiB
マシン 2	CPU	Intel Xeon X5680 @ 3.33GHz ×12
	メモリ	125.3GiB
	GPU	GeForce GTX 1080 Ti 11171MiB
共通	OS	Ubuntu 16.04 LTS（64bit）
	NN フレームワーク	Chainer 5.1.0
	エンコーダ	HM 16.20

データセット作成のための最適モード情報と MPM の導出は C++（HM）を用いて行い、その他の処理は python を用いて行う。図 4.5 に実験の実行環境を示す。CNN の学習の際、ミニバッチサイズは 100、最適化手法は Adam を使用する。学習の epoch（学習データを全て使い切るサイクル）数は、学習させたときの loss により決定する。図 4.6、図 4.7 に、それぞれ構造(1-1)を学習させたときの accuracy と loss を示す。両図より、50

epoch で十分収束していると判断し、学習の epoch 数は 50 とした。他の構造についても同様に判断し epoch 数は 50 とした。

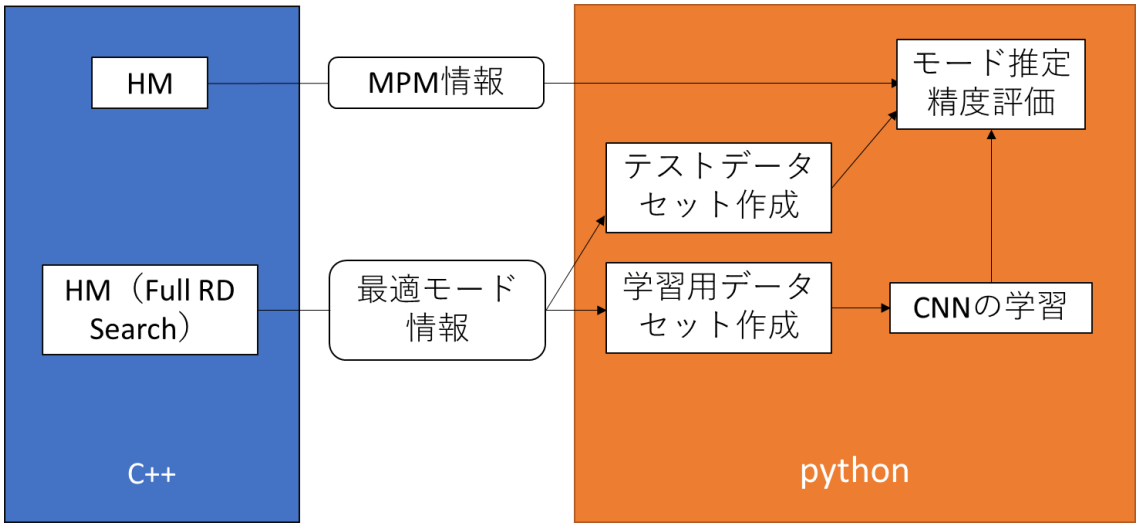

図 4.5　実験の実行環境

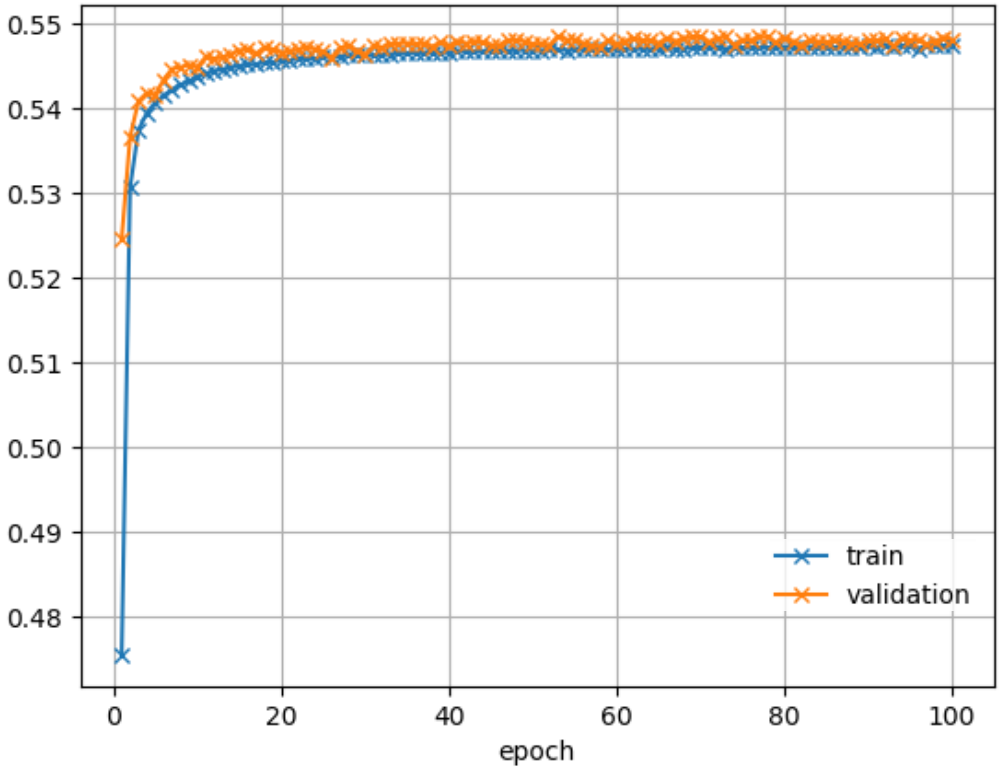

図 4.6　構造(1-1)を学習させたときの accuracy

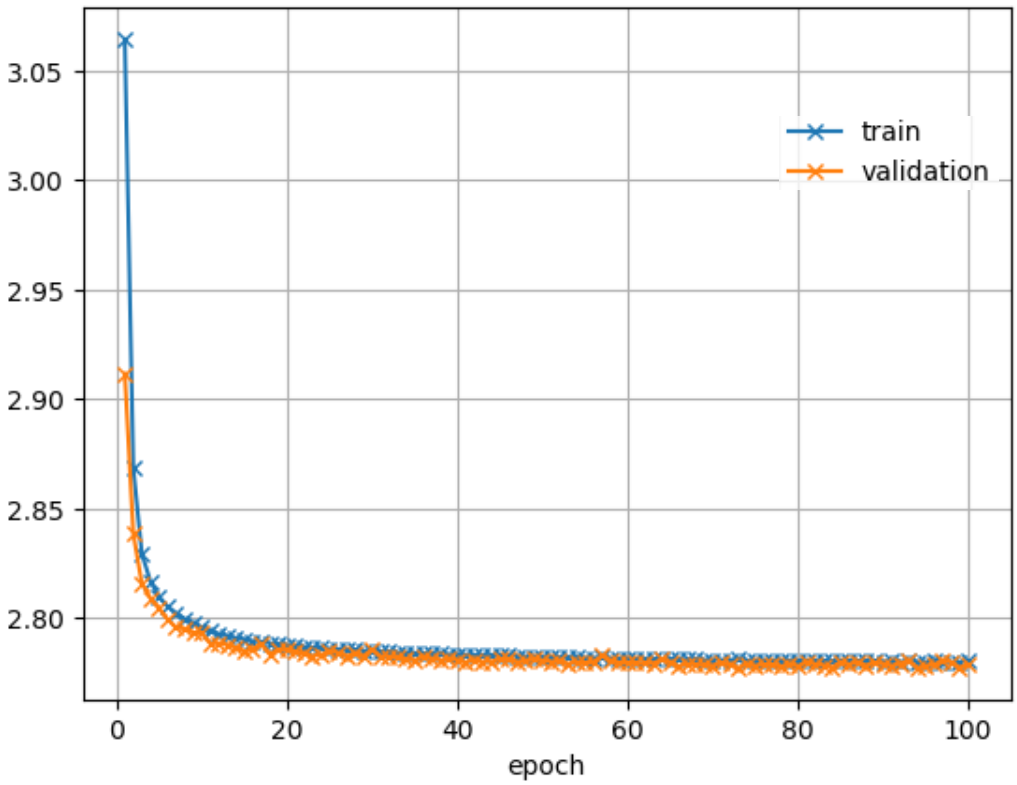

図 4.7　構造(1-1)を学習させたときの loss

4.2.3 実験結果

表 4.5 に、精度評価実験の結果を示す。精度は 0～1 の値で表されている。構造の違いによる精度の差はあまり大きくなく、いずれのモードの組み合わせでも数%程度の差となっている。下線の引かれた値は、モードの組み合わせ(i)すなわち MPM の 3 つのモードで推定した時の精度を上回った値を示す。最も推定精度が高いのは構造(3-2)、組み合わせ(h)の時に 72.3%で組み合わせ(i)より 10%以上高い精度になっている。一方、精度が上回った組み合わせは(e)～(h)で、これらは CNN からの出力モードと MPM あわせて 4 モード以上使用した組み合わせとなっている。この結果から、MPM を CNN から得た候補モードで置き換えても最適画面内予測モード推定の精度が向上しないことがわかる。

表 4.5　精度評価実験の結果

ネットワーク構造	推定に用いた CNN からの出力モードと MPM のモードの組み合わせ								
	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)
(1-1)	0.553447	0.482021	0.578662	0.581357	0.646133	0.64623	0.694414	0.722178	0.618164
(1-2)	0.551807	0.481152	0.578643	0.581582	0.644687	0.645303	0.693818	0.721406	
(1-3)	0.463223	0.472803	0.552383	0.581797	0.626846	0.608857	0.673105	0.704229	
(2-1)	0.553174	0.481543	0.580322	0.581416	0.644561	0.646699	0.694326	0.721748	
(2-2)	0.550283	0.481064	0.579795	0.581172	0.645244	0.645410	0.693467	0.720654	
(2-3)	0.522109	0.476523	0.56542	0.578867	0.642051	0.631514	0.686289	0.714736	
(3-1)	0.463223	0.472803	0.552549	0.581797	0.626875	0.608857	0.673105	0.704229	
(3-2)	0.555781	0.482588	0.58166	0.581904	0.645527	0.648271	0.695361	**0.723301**	
(3-3)	0.463154	0.472803	0.552686	0.581797	0.626826	0.608770	0.673037	0.704238	

第 5 章 提案手法

第 4 章で説明した CNNMC のネットワークではあまり高い精度で最適画面内予測モードを推定することができなかった。本章ではより高い精度で最適画面内予測モードを推定するネットワークとそれにより推定されたモードを用いて符号化を行う提案手法について述べる。

5.1 最適画面内予測モードの推定

本節では、提案手法のうち最適画面内予測モードを推定するために用いる CNN とその学習・評価用のデータセットについて述べる。

5.1.1 使用する CNN の概要

図 5.1 に、4 章で説明した CNNMC のネットワークと問題点を示す。CNNMC のネットワークは、128×128 の大きなブロックの情報を入力および出力する。そのため、128×128 のブロックの中に存在する様々な PU の画面内予測モードが一度に推定される。この時、図中に示しているようにブロックサイズが大きい影響で Reconstructed Block と PU が離れているケースが発生してしまう。Reconstructed Block と PU が離れるほど、最適画面内予測モードの相関が低くなり、Reconstructed Block の情報から PU の最適画面内予測モードを推定するのが難しくなると考えられる。

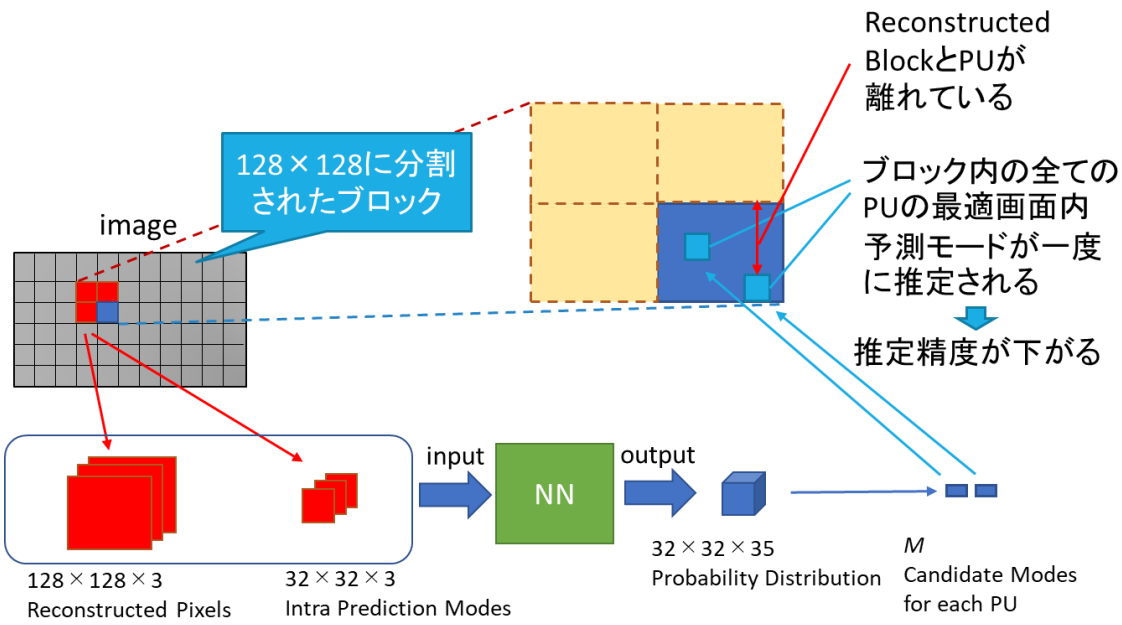

図 5.1　CNNMC のネットワークと問題点

以上で述べた問題点を解決するネットワークとして、以下のものを提案する。図 5.2 に提案するネットワークの概要を示す。提案するネットワークでは、Reconstructed Block として 128×128 の大きなブロックではなく、最適画面内予測モード推定対象の PU と同じ大きさの隣接する PU を使用する。PU のサイズは可変なのでネットワークに入力する Reconstructed Pixels のサイズもそれに合わせて変わる。そのためネットワーク構造も PU のサイズ 5 種類（4×4、8×8、16×16、32×32、64×64）に合わせて変更した 5 種類を用意し、それぞれに対応するデータセットを作成して学習させる。また、Intra Prediction Modes は左・左上・上で隣接する PU で最適モードとして使用された 3 つのモードの情報が格納された行列である。この行列は、ネットワークの出力が 35 種類の画面内予測モードの確率分布を格納したベクトルになっているのに合わせて、モードを one-hot 表現（モード番号とベクトルのインデックスを対応付け、該当モードに対応するベクトル要素を 1、そうでないものを 0 とする表現）にしたベクトル 3 つからなる行列となっている。図 5.3 に one-hot 表現の例を示す。

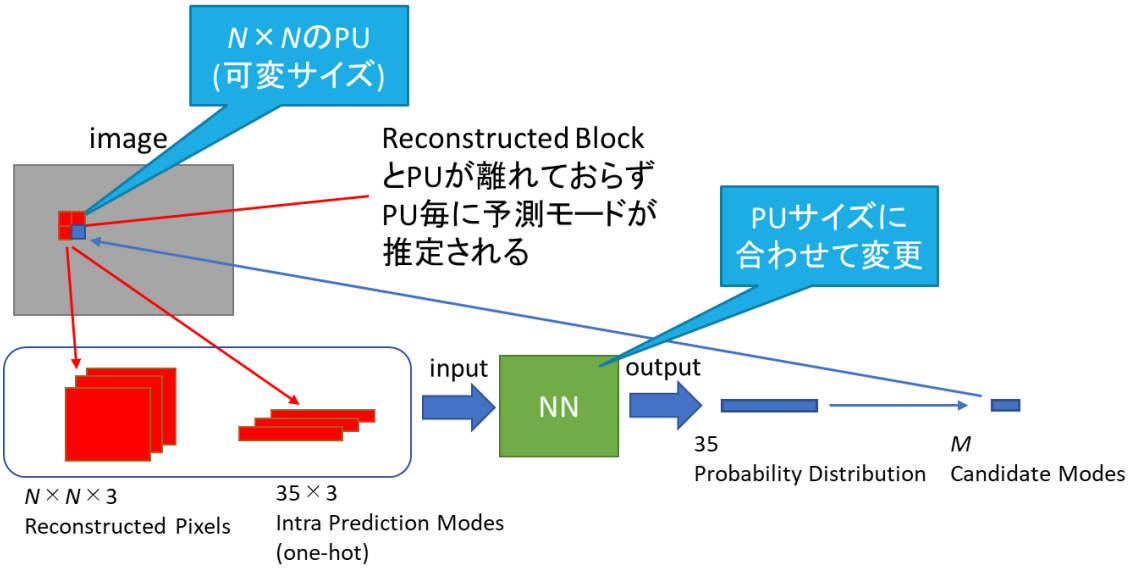

図 5.2　提案するネットワークの概要

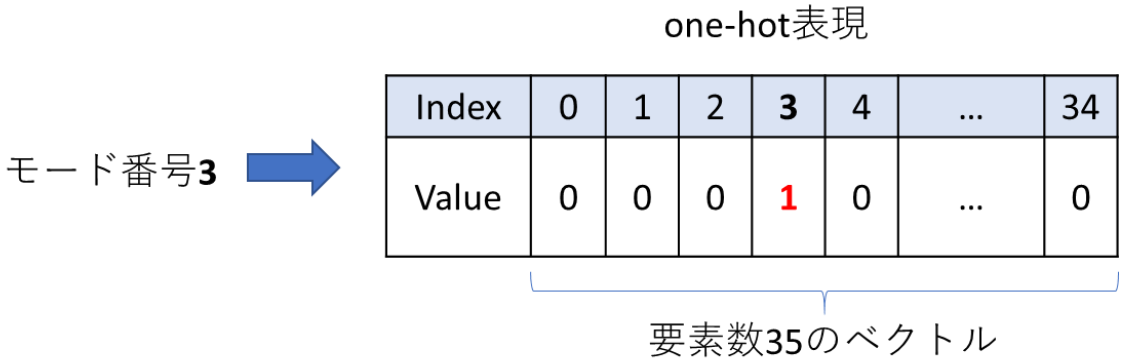

図 5.3　one-hot 表現の例

5.1.2 使用する CNN の構造

使用する CNN の構造は、[32][34]、4.2 節での結果を参考に構造(A)と(B)の 2 種類を検討する。図 5.4 と表 5.1 に構造(A)のネットワークとパラメータ、図 5.5 と表 5.2 に構造(B)のネットワークとパラメータを示す。構造(A)において、最大値 1 最小値 0 に正規化された Reconstructed Pixels の入力は 2 つの Convolution 層と 1 つの FC 層を、Intra Prediction Modes の入力は 1 つの FC 層を経て足し合わされた後、1 つの FC 層を経て Softmax 関数にかけられる。各 Convolution 層、FC 層の後には活性化関数として Tanh が用いられる。構造(B)は、構造(A)の conv1 層と conv2 層の後にそれぞれ Convolution 層

を追加した構造になっていて、PU サイズ（N）が 8 より大きければ Stride が 2、8 以下であれば Stride が 1 となる。

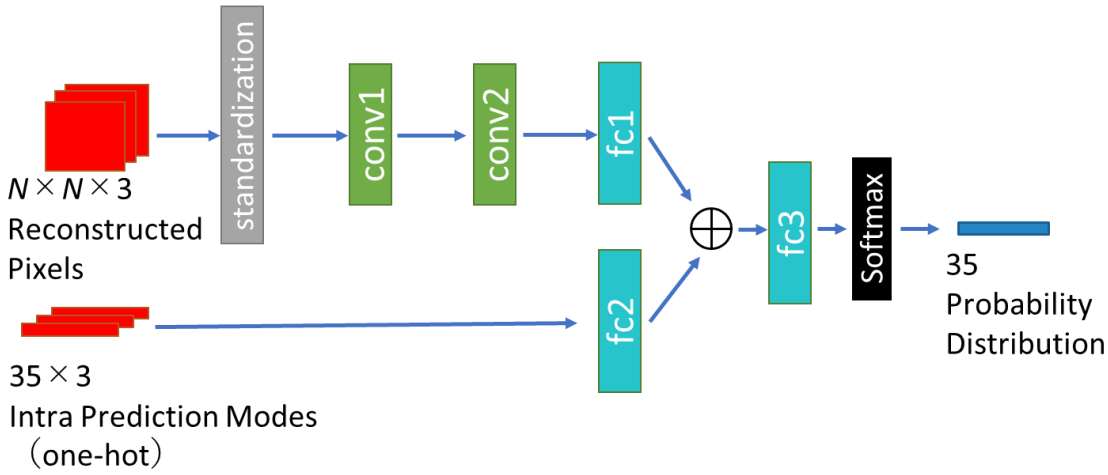

図 5.4　構造(A)のネットワーク

表 5.1　構造(A)で用いられるパラメータ

Layer	Kernel Size	Channel 数	Stride
conv1	3×3	128	1
conv2	3×3	64	1
fc1	-	128	-
fc2	-	128	-
fc3	-	35	-

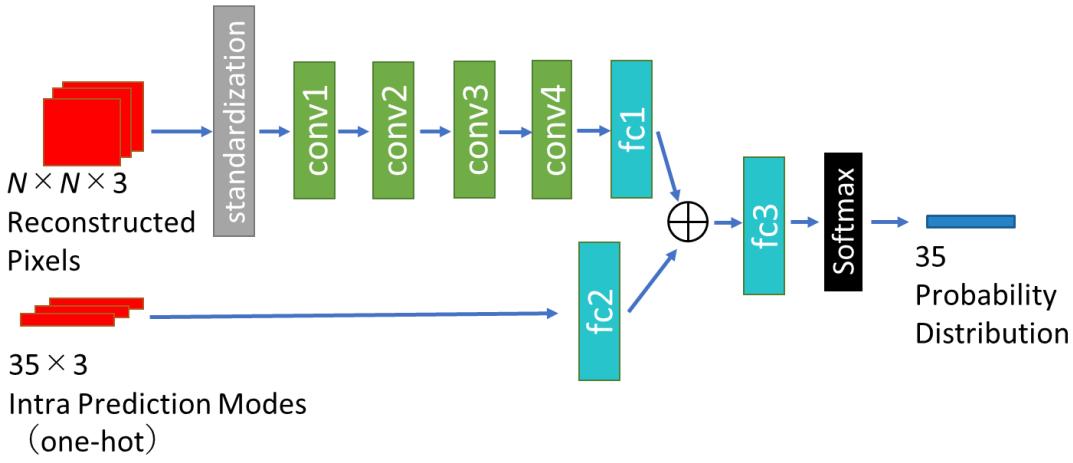

図 5.5　構造(B)のネットワーク

表 5.2　構造(B)で用いられるパラメータ

Layer	Kernel Size	Channel 数	Stride
conv1	3×3	128	1
conv2	3×3	128	2 (1)
conv3	3×3	64	1
conv4	3×3	64	2 (1)
fc1	-	128	-
fc2	-	128	-
fc3	-	35	-

( )内は N=4、8 の時

5.1.3 CNN の学習・評価用のデータセット

データセットは 4.2.1 節と同様の手順で作成する。4.2.1 節のデータセットと異なるのは、128×128 のブロック毎ではなく PU 毎に Reconstructed Pixels、Intra Prediction Modes、Target Modes の情報を取り出して組にする点、PU サイズ毎にデータセットを作成する点である。図 5.6 に提案するデータセット作成の概要を示す。

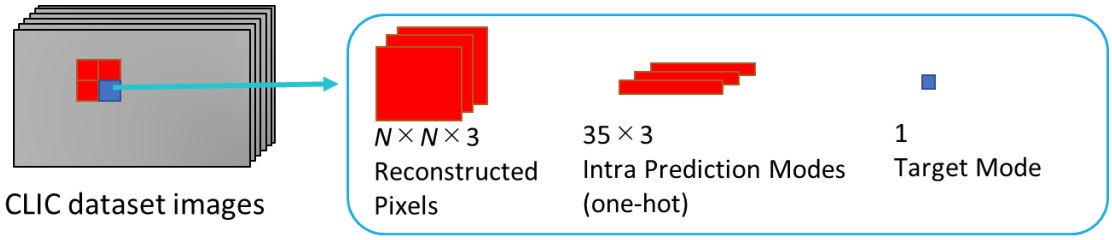

図 5.6　提案するデータセット作成の概要

5.2 推定された最適画面内予測モードを用いた符号化

本節では、提案手法のうち推定されたモードを用いた符号化について述べる。

提案手法では、3.3 節で示した HM における画面内予測モードの決定のフローのうち、MPM を導出して RD コスト計算対象とする部分を、CNN により推定された 3 つのモードを RD コスト計算対象とするように変更を加えて符号化を行う。図 5.7 に HM において提案手法を追加した最適画面内予測モード決定の流れを示す。

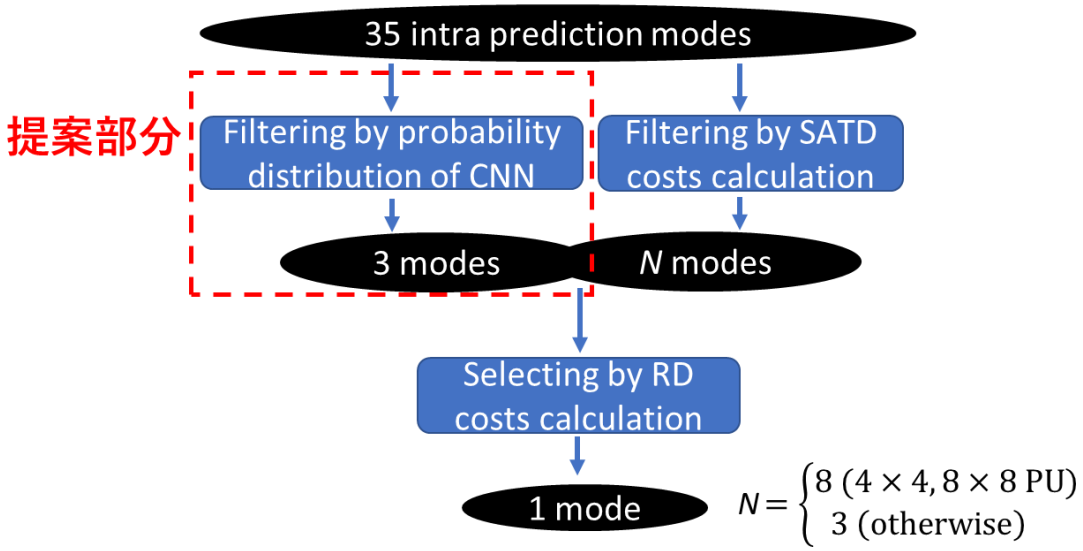

図 5.7　HM において提案手法を追加した最適画面内予測モード決定の流れ

第 6 章 評価実験

本章では、第 5 章で説明した提案手法の評価実験について述べる。はじめに実験環境について説明した後、最適画面内予測モードの推定精度評価実験と、推定された最適画面内予測モードを用いた符号化性能評価実験について述べる。

6.1 実験環境

4.2.2 節で述べた実験環境に加えて以下の実験環境を使用する。表 6.1 に新たに加えた計算環境を示す。

表 6.1　実験に使用した計算環境

マシン 3	CPU	Intel Core i7-3770 @ 3.40GHz ×4
	メモリ	16.0GB
	OS	Windows 10 Pro（64bit）
	NN フレームワーク	ONNX Runtime 1.1.0
	画像処理ライブラリ	OpenCV 3.4.8
共通	エンコーダ	HM 16.20

データセットの作成のための最適モード情報導出と符号化性能評価は C++（HM）を用いて行い、その他の処理は python を用いて行う。表 6.1 のマシン 3 を用いたのは符号化性能評価のときのみである。図 6.1 に実験の実行環境を示す。CNN の学習の際、ミニバッチサイズは 100、最適化手法は Adam を使用する。学習の epoch 数は、学習させたときの loss により決定する。図 6.2、図 6.3 に、それぞれ 4×4 PU の最適モードを構造(A)に学習させた時の accuracy と loss を示す。両図より、50 epoch で十分収束していると判断し、学習の epoch 数は 50 とした。他の PU サイズ、構造についても同様に判断し epoch 数は 50 とした。CNN の学習は python の Chainer で行うが、符号化性能評価時に C++（HM）の中で推論処理を行う必要があるため、それが可能な ONNX（Open Neural Network Exchange）[35]形式のモデルに変換する。変換には ONNX-Chainer を用い、推論には ONNX Runtime を用いる。推論の際は GPU ではなく CPU を用いる。

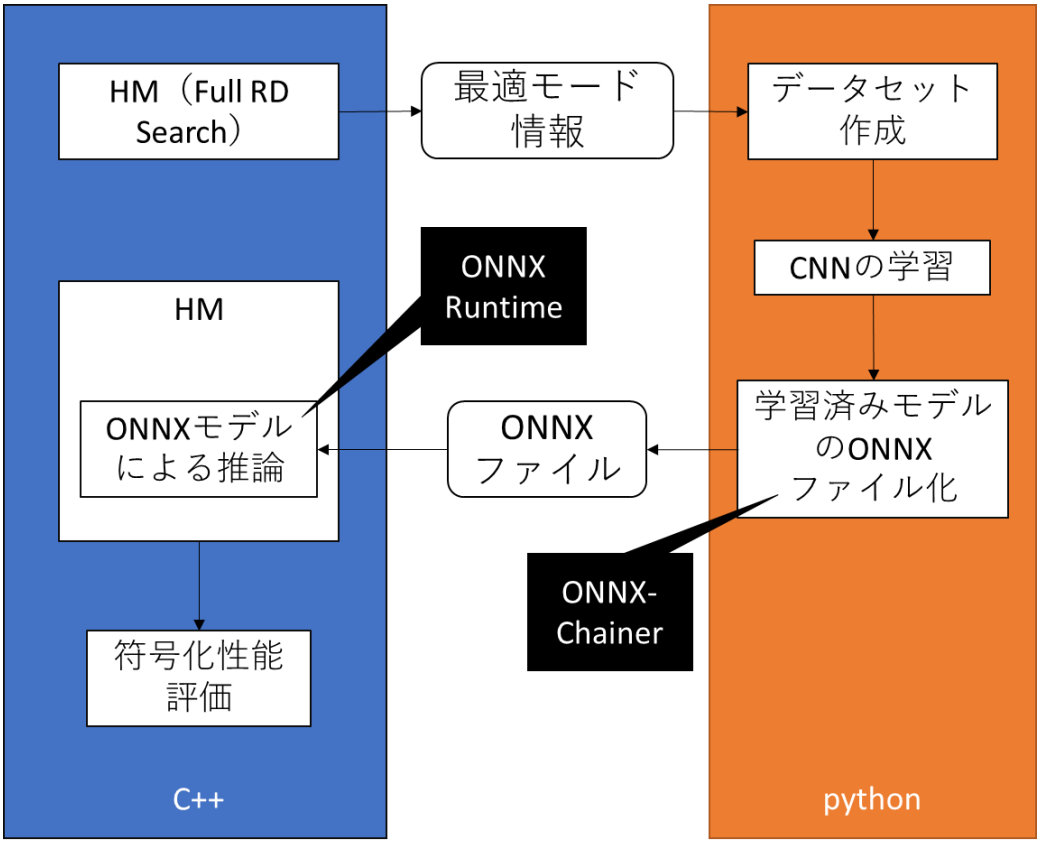

図 6.1　実験の実行環境

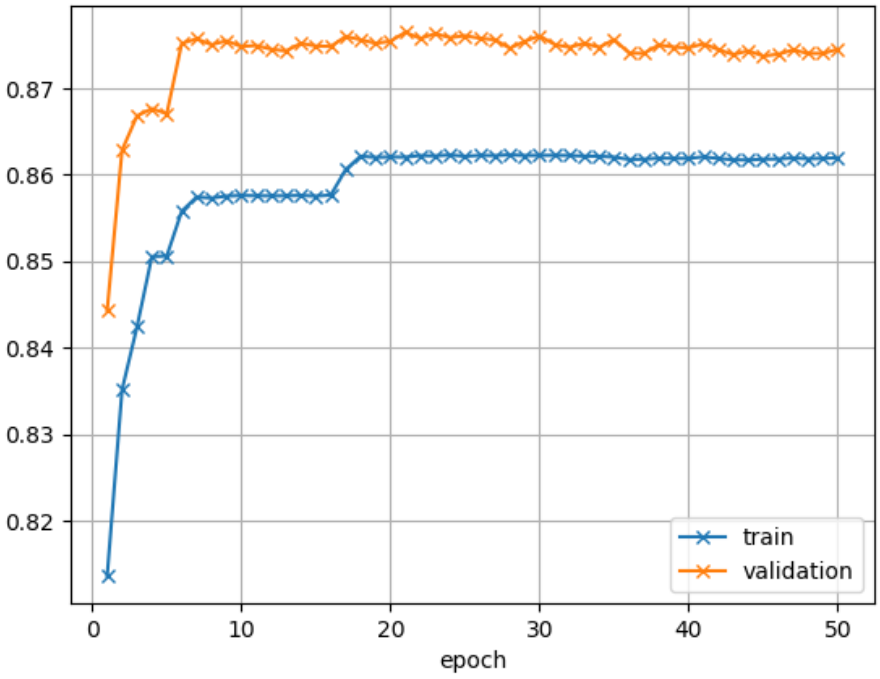

図 6.2　4×4 PU の最適モードを構造(A)に学習させた時の accuracy

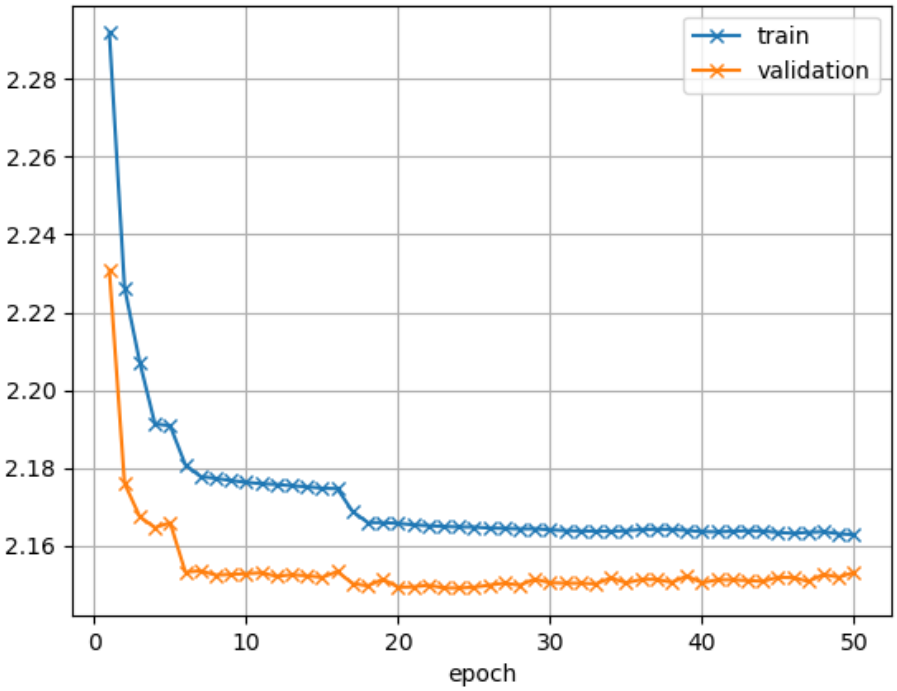

図 6.3　4×4 PU の最適モードを構造(A)に学習させた時の loss

6.2 最適画面内予測モードの推定精度評価実験

図 6.4 に、最適画面内予測モードの推定精度評価実験の概要を示す。この実験では、テスト用データセットの Reconstructed Pixels と Intra Prediction Modes を学習した CNN に入力して得られた 3 つの候補モードの中に Target Mode が含まれる割合を算出して最適画面内予測モード推定精度の評価を行う。

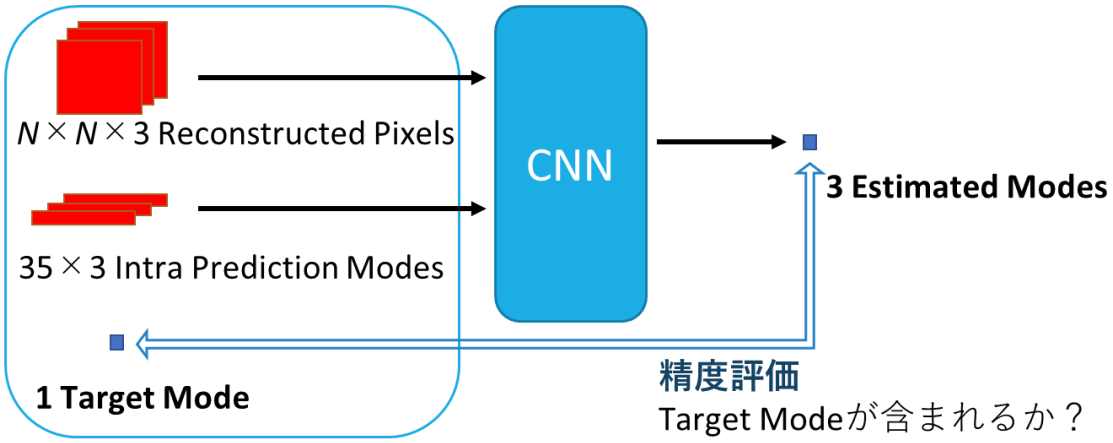

図 6.4　最適画面内予測モードの推定精度評価実験の概要

精度評価にあたり、4.2 節で示した CNNMC、MPM の精度とも比較を行う。なお、4.2 節のデータセットが PU サイズ 4×4 の時を対象としているため、PU サイズが 4×4 の時の精度比較となる。

図 6.5 に、各手法の最適画面内予測モードの推定精度を示す。同図より、提案手法は他の 2 つの手法の精度を上回ることが確認できる。そして、構造(A)を用いるほうが構造(B)を用いるよりもわずかに精度が高く、MPM と比較すると 25.6%高い精度になっている。

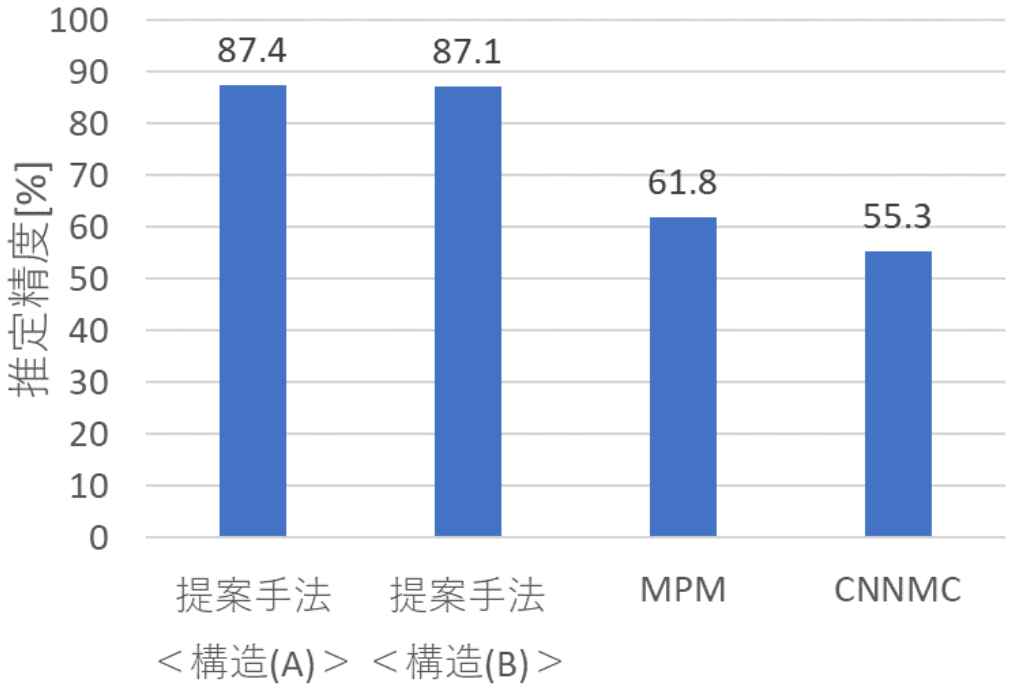

図 6.5　各手法の最適画面内予測モードの推定精度

次に、提案手法において PU サイズ毎の推定精度の比較を行う。

図 6.6 に、PU サイズ毎の最適画面内予測モードの推定精度を示す。PU サイズが 4×4、8×8 の時は構造(A)を用いる方が高い精度となり、16×16 以上の時は構造(B)を用いる方が高い精度となっている。全サイズの平均をとると、小さい PU の方が大きい PU よりも数が多いため、構造(A)の精度が構造(B)の精度を上回る結果となっている。

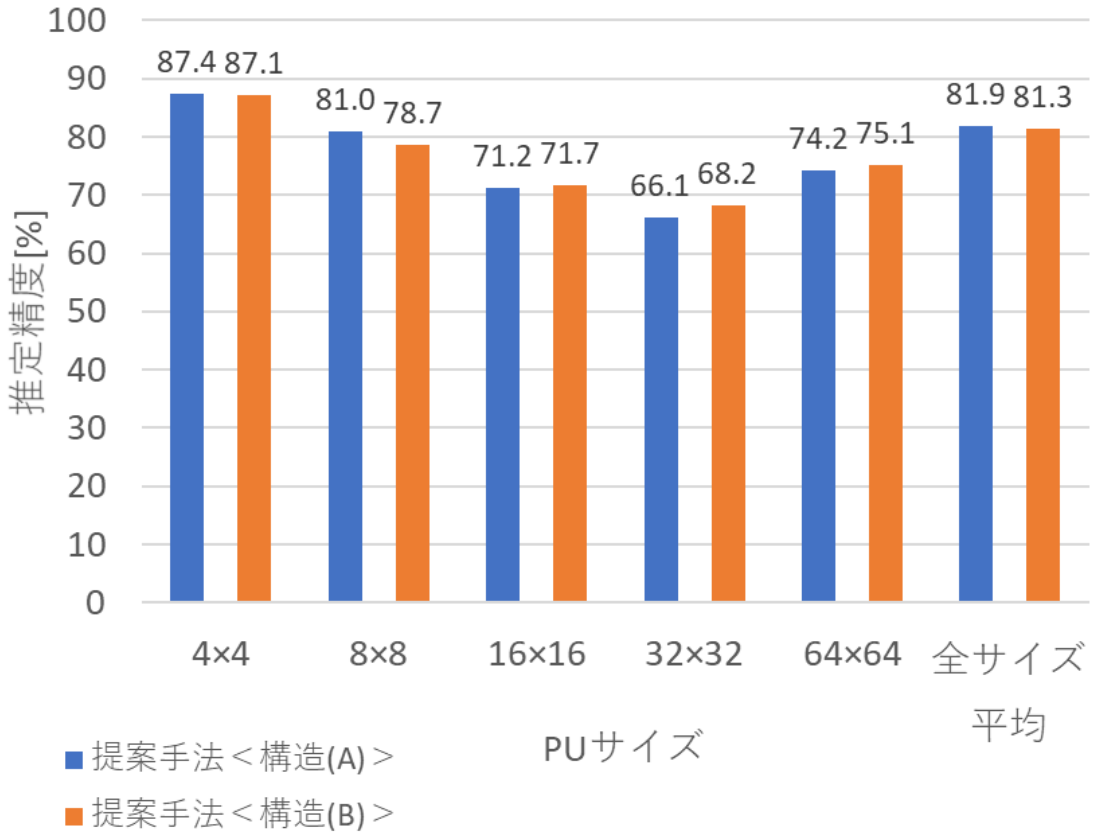

図 6.6　PU サイズ毎の最適画面内予測モードの推定精度

6.3 推定された最適画面内予測モードを用いた符号化性能評価実験

前節では、提案手法が MPM よりも高い精度で最適画面内予測モードが推定できることを確認した。本節では、提案手法により推定された最適画面内予測モードを用いて符号化を行い、従来手法の HM と性能を比較する。HM はこれまでと同様に All Intra で符号化を行う。テスト画像には、これまでテスト用データセットに使用してきた CLIC dataset の validation 用画像を使用する。また、Full RD Search を行う場合とも比較を行う。性能評価にあたって、HM の Common Test Conditions（CTC）[36]に従って QP＝22、27、32、37 として各画像符号化を行い、それぞれの手法の HM に対する BD-Rate を算出して平均する。

表 6.2 に平均 BD-Rate の比較結果を示す。提案手法はいずれも Y の BD-Rate が下がり、HM よりも良い結果となっている。特に構造(A)は構造(B)の 5 倍以上 BD-Rate が向上している。Full RD Search と比較すると、Y の BD-Rate では劣るものの U と V の BD-

Rate では提案手法の方が良い結果となっている。特に構造(A)は U の BD-Rate の悪化の度合いが低く、V の BD-Rate は向上している。

表 6.2　平均 BD-Rate の比較結果

	平均 BD-Rate [%]		
	Y	U	V
提案手法＜構造(A)＞	－0.163	0.013	－0.137
提案手法＜構造(B)＞	－0.031	0.109	0.044
HM（Full RD Search）	－0.311	0.188	0.076

次に、各手法の BD-Rate（Y）の分布を比較する。図 6.7 に各手法の BD-Rate の分布を示す。この図は、BD-Rate を 0.5%ずつ刻んだ区間に該当する画像の数を求めヒストグラムにしたものとなっている。提案手法は表 6.2 に示した平均値付近に分布が集中しているのに対し、Full RD Search は平均値付近だけではなく低い BD-Rate の方にも少し分布があり、ややばらつきがあるのが確認できる。構造(A)、Full RD Search とも-0.2%前後に多くの画像が分布しているが、一部の画像では Full RD Search の BD-Rate が低くなっていて構造(A)との BD-Rate の差が大きくなることがわかる。そしてその差が平均 BD-Rate の差として表れていると考えられる。

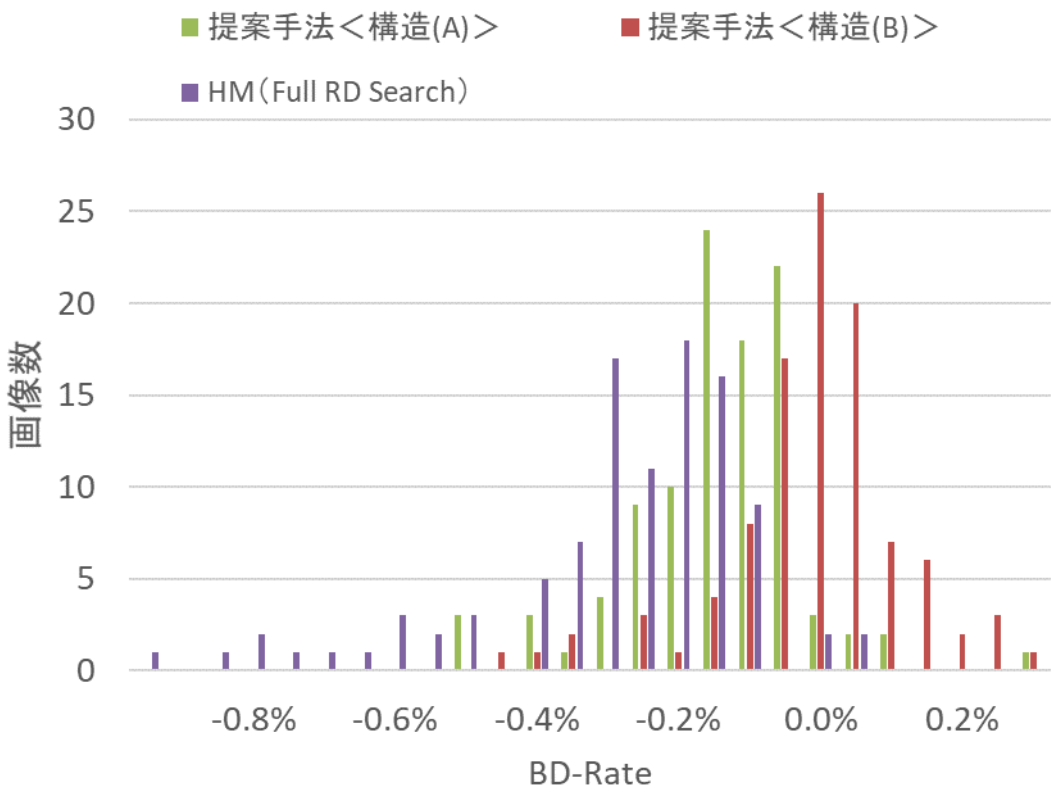

図 6.7　各手法の BD-Rate の分布

続いて、PSNR とビット数の分布をみていく。横軸を HM とのビット数の比、縦軸を HM との PSNR の差として、各画像をエンコードした時の相対ビット数と相対 PSNR をプロットしたグラフを作成する。QP＝22 の高ビットレートの場合と QP＝37 の低ビットレートの場合について比較を行う。

図 6.8 に QP＝22 の時、図 6.9 に QP＝37 の時の各手法の相対 PSNR と相対ビット数の分布を示す。なお両図においては、上に行くほど PSNR があがり左に行くほどビットレートが下がるため、左上に行くほど符号化効率が向上することとなる。

図 6.8 を見ると、高ビットレートの時は Full RD Search が群を抜いて高い PSNR となり、次いで構造(A)が構造(B)よりわずかに高い PSNR となる傾向が見える。Full RD Search は PSNR が上がる分ビット数も増えている。

図 6.9 を見ると、低ビットレートの時は高ビットレートの時ほど Full RD Search の PSNR は高くないことが確認できる。特に構造(A)の分布と Full RD Search の分布が高ビットレートの時と比べて大きく重なり合っており、低ビットレートの時は符号化効率の差が小さくなることがわかる。

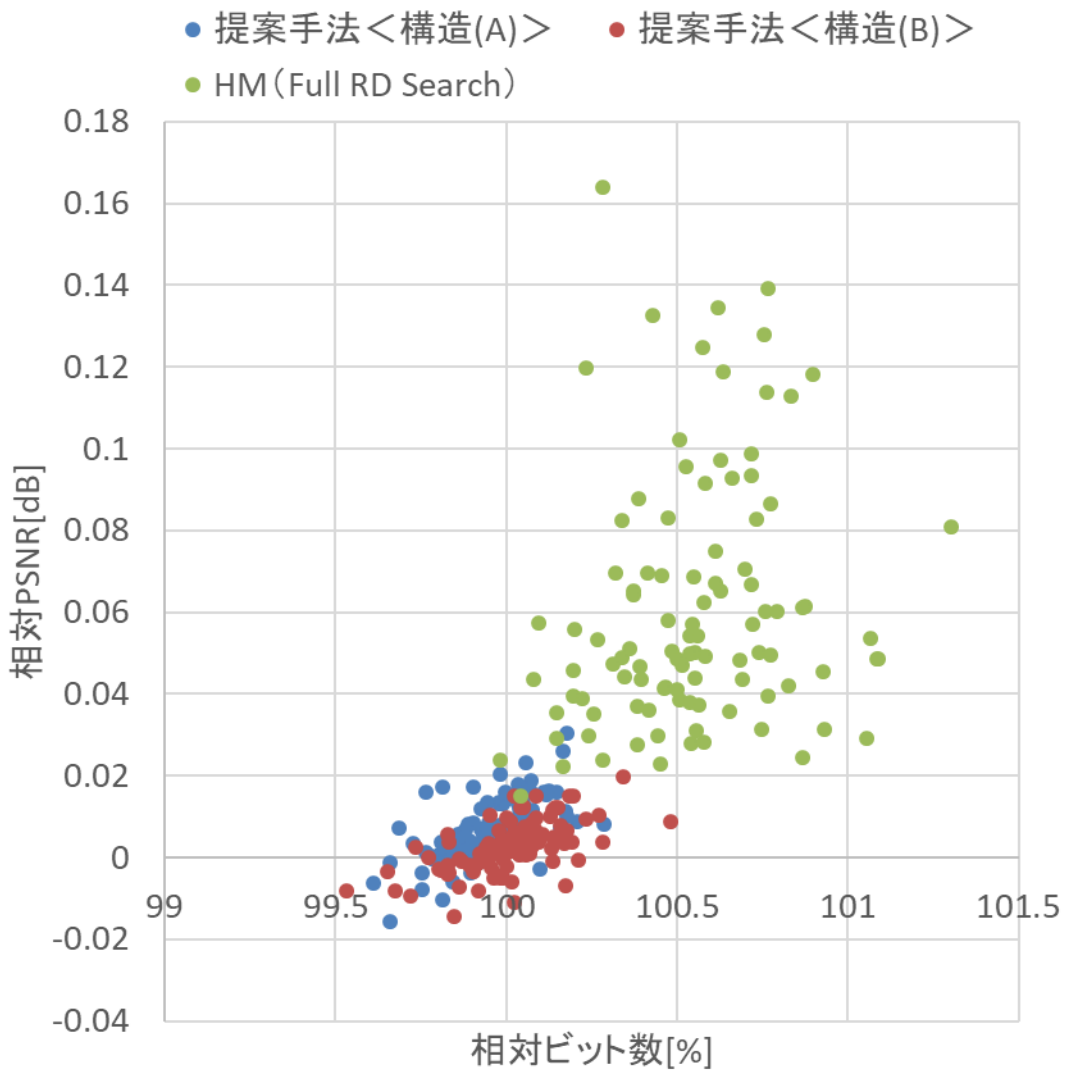

図 6.8　QP＝22 の時の各手法の相対 PSNR と相対ビット数の分布

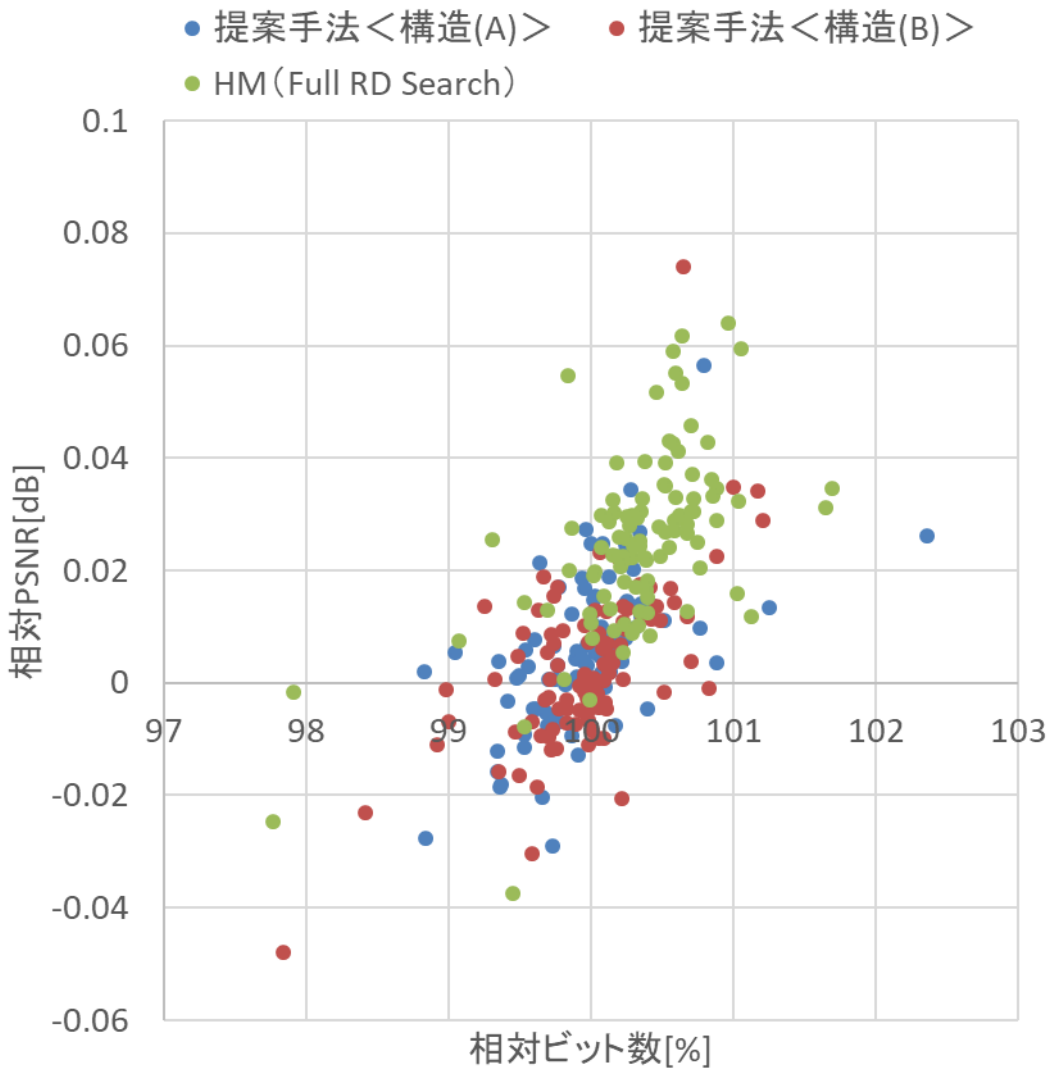

図 6.9　QP＝37 の時の各手法の相対 PSNR と相対ビット数の分布

最後に、各手法を HM とエンコード時間とデコード時間で比較する。テスト画像は先ほど使ったものの中からランダムに 10 枚選んでエンコード、デコードを行う。

表 6.3 に、各手法の HM に対するエンコード時間とデコード時間を示す。エンコード時間はどの手法も増加している。構造(A)と Full RD Search のエンコード時間はあまり大きな差がないが、構造(B)は他 2 つと比べて 2 倍以上エンコード時間がかかっている。デコード時間は Full RD Search がオリジナルの HM とほとんど変わらない一方で、提案手法は大幅に増加する結果となっている。

表 6.3　各手法の HM に対するエンコード時間とデコード時間

	エンコード時間	デコード時間
提案手法＜構造(A)＞	+582%	+7102%
提案手法＜構造(B)＞	+250%	+1854%
HM（Full RD Search）	+212%	-2%

第 7 章 総括

7.1 まとめ

本論文では、HEVC の参照ソフトウェアである HEVC Test Model（HM）において、CNN を用いてより RD 最適な画面内予測モードを選択して符号化を行う手法を提案した。提案手法において、2 つの異なる数の層を持つ CNN を検討した。いずれの構造の CNN を用いた場合も、最適画面内予測モード推定の精度が従来の MPM や CNNMC を用いた場合の精度を上回る結果となった。さらに、CNN により推定された最適画面内予測モードを用いて符号化を行い、性能評価を行った。BD-Rate などにより従来手法の HM と比較した結果、符号化効率の改善が示された。また、層の数が少ない浅い CNN を用いた方が高い符号化効率となることも示された。Full RD Search と比べて符号化効率が及ばなかったものの、低ビットレートの時は符号化効率の差が小さくなることもわかった。

7.2 今後の展望

提案手法は、従来の HM と比較して符号化効率が向上したもののエンコードとデコードにかかる時間が大幅に増加する結果となった。今後は GPU による効率の良い演算などにより演算時間の短くなるような実装や手法を検討する必要がある。また、標準化が間もなく行われる VVC の参照ソフトウェア VVC Test Model（VTM）へ応用することも考えている。

謝辞

本研究を行うにあたり、様々なご指導をしてくださった甲藤二郎教授に心から感謝いたします。また、研究を進める上での貴重なアドバイスなど様々な面においてお世話になった孫鶴鳴さん、竹内健さん、金井謙治さんをはじめとする甲藤研究室の皆さま、松尾康孝さんをはじめとする NHK 放送技術研究所の皆さまに厚く御礼申し上げます。最後に、これまで私を育て支えてくださった家族や友人、多くの方々に深く感謝いたします。

2020 年 1 月 29 日

横山　怜汰

参考文献

[1] 総務省, “情報通信白書平成 28 年版,” 2016.

[2] Cisco, “Cisco Visual Networking Index（VNI）: 予測とトレンド、2017 ～ 2022 年 ホワイト ペーパー,” 2018.

[3] 村上伸一, “画像通信工学,” 東京電機大学出版局, 1994.

[4] 神奈川工科大学 情報学部 情報工学科 信号処理応用研究室, “標準画像／サンプルデータ,” http://www.ess.ic.kanagawa-it.ac.jp/app_images_j.html, 2020 年 1 月 19 日閲覧.

[5] Keith Jack, “Communications Engineering Desk Reference - Chapter 7.2 Colour spaces,” Academic Press, pp. 469-482, 2009.

[6] 山岸秀一, “知っておきたいキーワード 4:4:4 と 4:2:0,” 映像情報メディア学会誌 Vol. 62, No. 10, pp. 1542-1546, 2008.

[7] 松田一朗, “フレーム内予測,” 映像情報メディア学会誌 Vol. 67, No. 3, pp. 240-243, 2013.

[8] 宮田高道, “予測符号化,” 電子情報通信学会「知識ベース」2 群－5 編－4 章, 2013.

[9] 橋本秀雄, “画像符号化アルゴリズム II—変換符号化—,” テレビジョン学会誌 Vol. 43, No. 10, pp. 1145-1155, 1989.

[10] 村上篤道, 浅井光太郎, 関口俊一, “高効率映像符号化技術 HEVC/H.265 とその応用,” オーム社, 2013.

[11] MathWorks, “ドキュメンテーション—離散コサイン変換,” https://jp.mathworks.com/help/images/discrete-cosine-transform.html, 2020 年 1 月 20 日閲覧.

[12] 長谷川まどか, “可変長符号化,” 映像情報メディア学会誌 Vol. 67, No. 1, pp. 41-45, 2013.

[13] 大久保榮, 鈴木輝彦, 髙村誠之, 中條健, “H.265/HEVC 教科書,” インプレスジャパン, 2013.

[14] 市ヶ谷敦郎, “次世代映像符号化方式の標準化動向,” NHK 技研 R&D No.177, 2019.

[15] Frank Bossen, Xiang Li, Karsten Suehring, “AHG report: Test model software development (AHG3),” JVET-Q0003-v1, 2019.

[16] Gisle Bjontegaard, “Calculation of average PSNR differences between RD-curves,” VCEG-M33, 2001.

[17] Zhou Wang, Alan Conrad. Bovik, Hamid Rahim Sheikh, Eero P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Trans. on Consumer Electronics vol.55, pp.210-217, 2005.

[18] 杉本修, “符号化画質,” 電子情報通信学会「知識ベース」2 群－5 編－9 章, 2013.

[19] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu, “Wavenet: A generative model for raw audio,” arXiv:1609.03499, 2016.

[20] Yoon Kim, “Convolutional neural networks for sentence classification,” Proc. EMNLP, pp.1746-1751, 2014.

[21] 内田祐介, 山下隆義, “物体認識のための畳み込みニューラルネットワークの研究動向,” 電子情報通信学会論文誌 D Vol. J102–D, No. 3, pp. 203–225, 2019.

[22] Vincent Dumoulin, Francesco Visin, “A guide to convolution arithmetic for deep learning,” arXiv:1603.07285v2, 2018.

[23] Yann LeCun, Leon Bottou, Yoshua Bengio, Patrick Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, Vol.86, No.11, pp.2278-2324, 1998.

[24] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” NIPS, 2012.

[25] Karen Simonyan, Andrew Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” ICLR, 2015.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition,” CVPR, 2016.

[27] Md Zahangir Alom, Tarek M. Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C. Van Essen, Abdul A. S. Awwal, Vijayan K. Asari, “A State-of-the-Art Survey on Deep Learning Theory and Architectures,” MDPI, 2019.

[28] Diederik P. Kingma, Jimmy Lei Ba, “Adam: A Method for Stochastic Optimization,” ICLR, 2015.

[29] John Duchi, Elad Hazan, Yoram Singer, “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization.” Journal of Machine Learning Research 12, pp.2121-2159, 2011.

[30] 斎藤康毅, “ゼロから作る Deep Learning,” オーム社, 2017.

[31] 鈴木輝彦, “フレーム内符号化とフレーム間予測符号化,” 映像情報メディア学会誌 Vol. 67, No. 7, pp. 537-540, 2013.

[32] Zhenzhong Chen, Yiming Li, Feiyang Liu, Zizheng Liu, Xiang Pan, Wanjie Sun, Yingbin Wang, Yan Zhou, Han Zhu, “CNN-Optimized Image Compression with Uncertainty based Resource Allocation,” IEEE CVPR Workshop, 2018.

[33] “Workshop and Challenge on Learned Image Compression (CLIC),” https://www.compression.cc/2018/, 2020 年 1 月 24 日閲覧.

[34] Thorsten Laude, Jorn Ostermann, “Deep learning-based intra prediction mode decision for HEVC,” PCS, 2016.

[35] “Open Neural Network Exchange,” https://onnx.ai/, 2020 年 1 月 24 日閲覧.

[36] Frank Bossen, “Common test conditions and software reference configurations,” JVET-L1100, 2013.

発表文献リスト

国際学会

[1] **Ryota Yokoyama**, Masahiko Tahara, Masaru Takeuchi, Heming Sun, Yasutaka Matsuo, Jiro Katto, “CNN Based Optimal Intra Prediction Mode Estimation in Video Coding,” IEEE ICCE, Jan. 2020.

[2] Masaki Yasumaru, **Ryota Yokoyama**, Zhengxue Cheng, Kenji Kanai, Jiro Katto, “Accuracy evaluations of contact-free heart rate measurement methods using 4K facial images,” IEEE ICCE, Jan. 2019.

[3] Yusuke Sakamoto, **Ryota Yokoyama**, Masaru Takeuchi, Yasutaka Matsui, Jiro Katto, “Improvement of H.265/HEVC Encoding for 8K UHDTV by GOP Size and Prediction Mode Selection,” IEEE ICCE, Jan. 2019.

[4] Masaru Takeuchi, Yusuke Sakamoto, **Ryota Yokoyama**, Heming Sun, Yasutaka Matsuo, Jiro Katto, “A Gamut-extension Method Considering Color Information Restoration Using Convolutional Neural Networks,” IEEE ICIP, Sep. 2019.

国内学会

[1] **横山怜汰**, 竹内健, 金井謙治, 甲藤二郎, “不審行動検出に向けた複数人物追跡とバイオセンシングの検討,” 電子情報通信学会 画像工学研究会, 2017 年 3 月.

[2] **横山怜汰**, 坂本悠輔, 竹内健, 松尾康孝, 甲藤二郎, “Neural Network を用いた Intra/Inter モード切替による H.265/HEVC 符号化,” 2018 年画像符号化シンポジウム/映像メディア処理シンポジウム（PCSJ/IMPS2018）, 2018 年 11 月.

[3] **横山怜汰**, 田原雅彦, 孫鶴鳴, 竹内 健, 松尾康孝, 甲藤二郎, “CNN を用いた動画像符号化における最適 Intra 予測モード推定,” 2019 年画像符号化シンポジウム/映像メディア処理シンポジウム（PCSJ/IMPS2019）, 2019 年 11 月.

[4] **横山怜汰**, 田原雅彦, 孫鶴鳴, 竹内 健, 松尾康孝, 甲藤二郎, “CNN による最適 Intra 予測モード推定を用いた動画像符号化,” 電子情報通信学会 画像工学研究会, 2020 年 2 月（発表予定）.

[5] 安丸昌輝, **横山怜汰**, 程正雪, 金井謙治, 甲藤二郎, “UHD 顔画像を用いた非接触心拍計測法の精度評価,” 2018 年画像符号化シンポジウム/映像メディア処理シンポジウム（PCSJ/IMPS2018）, 2018 年 11 月.

[6] 安丸昌輝, **横山怜汰**, 程正雪, 金井謙治, 甲藤二郎, “4K 顔映像を活用した非接触心拍推定法の精度評価,” 情報処理学会 オーディオビジュアル複合情報処理研究会, 2018 年 11 月.

[7] 坂本悠輔, **横山怜汰**, 竹内 健, 松尾康孝, 甲藤二郎, “空間と動きの事前解析を用いた 8K 映像の H.265/HEVC 符号化における GOP サイズ推定,” 電子情報通信学会 画像工学研究会, 2019 年 3 月.

[8] 安丸昌輝, **横山怜汰**, 程正雪, 金井謙治, 甲藤二郎, “様々な実験環境下における 4K カメラを活用した非接触型心拍推定手法の精度評価,” 電子情報通信学会 通信方式研究会, 2019 年 3 月.

[9] 坂本悠輔, **横山怜汰**, 竹内健, 松尾康孝, 甲藤二郎, “事前の画像解析を用いた 8K 映像の H.265/HEVC 符号化における GOP サイズおよび intra/inter モード切替の推定,” 電子情報通信学会総合大会, 2019 年 3 月.

[10] 竹内健, **横山怜汰**, 孫鶴鳴, 松尾康孝, 甲藤二郎, “畳み込みニューラルネットワークを用いた色彩情報復元を考慮した色域拡張手法,” 2019 年画像符号化シンポジウム/映像メディア処理シンポジウム（PCSJ/IMPS2019）, 2019 年 11 月.

[11] 田原雅彦, **横山怜汰**, 孫鶴鳴, 松尾康孝, 甲藤二郎, “VVC における参照画素補間フィルタに関する検討,” 2019 年画像符号化シンポジウム/映像メディア処理シンポジウム（PCSJ/IMPS2019）, 2019 年 11 月.